PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-049545
(43)Date of publication of application : 20.02.1998

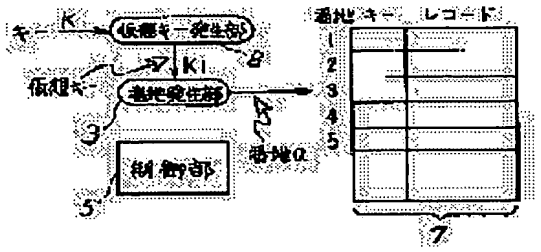
(51)Int.Cl. G06F 17/30
G11C 15/00

(21)Application number : 08-206971
(22)Date of filing : 06.08.1996

(71)Applicant : FUJITSU LTD
(72)Inventor : MINAMI TOSHIAKI

(54) ASSOCIATIVE STORAGE DEVICE

(57)Abstract:
PROBLEM TO BE SOLVED: To reduce partiality of data and a decrease conflicts by generating an address of a table on the basis of a virtual key and making access to an entry possible with the generated address.
SOLUTION: This device is equipped with a virtual key generation part 8, an address generation part 3, a table 7, and a control part 5, and in the table 7, an entry consisting of one key and a record corresponding to it is made to correspond to each address. This device registers and retrieve data (composed of a key and a record) to each entry of the table 7 under the control of the control pat 5, but the virtual key generation part 8 generates a virtual key in this case from a given key by a hash function and the address generation part 3 generates an address on the basis of the virtual key. Then, each entry of the table 7 is accessed with the generated address to register and retrieve data.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

CLAIMS

[Claim(s)]

[Claim 1] The table on which it had two or more entries which consist of a group of the record corresponding to one key and it, and the address was shaken at each entry, The virtual key generating section which generates a virtual key using a Hash Function based on the given key, Associative storage characterized by enabling access to said each entry with the address which was equipped with the address generating section which generates the address of said table based on the virtual key generated in said virtual key generating section, and was generated using the hashing method.

[Claim 2] Associative storage according to claim 1 characterized by having the address re-calculation section which re-calculates an address based on the virtual key generated in said virtual key generating section when an address is generated in said address generating section and a collision takes place.

[Claim 3] Associative storage according to claim 1 characterized by having the tree structure generation control means which generates the tree structure using said link by the size comparison of said virtual key when an address is generated in said address generating section and a collision takes place, while establishing the link of two or more branching in each entry of said table.

[Claim 4] The table on which it had two or more entries which consist of a group of one key, the standard key corresponding to it, and a record, and the address was shaken at each entry, The standard key generating section which generates a standard key using a Hash Function based on the given key, The virtual key generating section which generates a virtual key using a Hash Function based on the standard key generated in said standard key generating section, Associative storage characterized by enabling access to said each entry with the address which was equipped with the address generating section which generates the address of said table based on the virtual key generated in said virtual key generating section, and was generated using the hashing method.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Field of the Invention] This invention relates to the associative storage which used especially hashing technique for various computers about available associative storage. To the given key, associative storage is equipment with which the record related with it is searched, and is used for the general equipment which needs management of the symbol table of a computer language processor and the data associative retrieval in a database etc.

[0002]

[Description of the Prior Art] Hereafter, the conventional example is explained based on a drawing.

(1) : the explanatory view 15 of associative storage and hashing equipment is the explanatory view 1 of the conventional example, and, as for associative storage and B Fig., A Fig. shows hash equipment. Moreover, drawing 16 is the explanatory view 2 of the conventional example, and, as for open address method hash equipment and B Fig., A Fig. shows chain method hash equipment (in the case of a dichotomy tree).

[0003] For example, equipment (it is described as "hashing equipment" or "hash equipment" below) using the hashing method as a basic technique for associative storage implementation of the symbol table used by the processor of a computer language is widely used so that Mr. Seiichi Nishihara's description "the technique of hashing and application" (Information Processing Society of Japan, Vol, 21, No.9, Sep.1980) may see.

[0004] Associative storage is equipment with which the data (it is described as a "record" below) related with the key k are searched using the key (it is usually a character string) k given to data, as shown in A Fig. of drawing 15 . The information related if needed is added in said retrieval processing. The concept of associative retrieval equipment can be expressed as a table (table) 7 with Key k and a record.

[0005] for example, in the processor of computer language, in order to analyze language, to identifiers which appeared in the program, such as reserved word and a variable name, the mold needs to store various information, such as something, and it is necessary to use it to whether it is reserved word and a variable name In such a situation, from an identifier, in order to obtain the related information, associative storage is used.

[0006] By the way, hashing equipment is a kind of associative retrieval equipment. it was shown in B Fig. of drawing 15 -- as -- hashing equipment -- setting -- each entry (group of one key and the record

corresponding to it) of associative storage (it corresponds to a table 7) -- addresses (address) 1, 2, 3, 4, and 5 ... is shaken. For example, in the case of a table with m entries, it is possible that the address (address) of 0 to m-1 is shaken.

[0007] The address in which the retrieval in hashing equipment stores the record by the address generating section 3 from the key k given first is calculated. The value acquired as an address must be under or more 0m. The function which calculates an address from Key k is called a "Hash Function." The value which can be taken as a key k is expressed with "a key set", a call, and K. The set which can be taken as an address is expressed with "an address set", a call, and A. Therefore, Hash Function (it corresponds to internal function of the address generating section 3) h will be a function called h:K->A.

[0008] If the entry of the address calculated using the Hash Function is vacant, the record related with Key k and it is stored there. Also in the case of retrieval, address calculation is carried out by the approach more nearly same than the given key k, and it looks for an entry with the key k. Generally, since there are few entries of a table far than the number (it is the number of elements of Set K, and the case of infinity is included theoretically) of the key which may be recognized, the phenomenon in which the same address is calculated happens from a different key. This is called "a collision."

[0009] Some approaches are proposed as management when said collision takes place. As the approach, it is the approach called an open address method or a chain method. The open address method hash equipment using said open address method is shown in A Fig. of drawing 16 , and the chain method hash equipment using said chain method is shown in B Fig. of drawing 16 .

[0010] When the address of a table 7 is calculated with said open address method hash equipment from the key k to which the address generating section 3 was given and a collision takes place, it is the approach the address re-calculation section 4 calculates a new address for said calculated address to origin with directions of a control section 5. In count of this new address, a delta value is calculated by a certain approach at the original address, and a new address is usually calculated by adding it in many cases.

[0011] Moreover, with said chain method hash equipment, a link information is added to each entry of a table 7, and it ties to the entry prepared outside among tables 7. That is, if an address is calculated from the key k to which the address generating section 3 was given and a collision takes place, directions of a control section 5 will tie said link further using said key k. Moreover, the way of combining said open address method and

chain method was also proposed conventionally.

[0012] **2: Processing explanation of open address method hash equipment ... Drawing 17 reference drawing 17 is the processing flow chart of conventional open address method hash equipment. Hereafter, based on drawing 17 , processing of open address method hash equipment is explained. In addition, S1-S6 show each processing step.

[0013] In this processing, if Key k is given from the exterior (S1), the address generating section 3 will use the given key k, and will calculate Variable a by Hash Function h (k). And the address generating section 3 generates the address (this is only hereafter described as "Address a") to which said variable a points (S2).

[0014] Next, if {table(a) as for which the entry of Address a is vacant judged whether it was empty ?} (S3), and the control section 5 is vacant, the record corresponding to Key k and it will be registered into the entry of Address a (S6), and registration will be ended. However, when the entry of Address a is not vacant, a control section 5 compares and judges {key(a) =k?} in accordance with the key k to which the key registered into Address a was given (S4).

[0015] Consequently, if in agreement, since a key will be registered, processing is ended. However, a key is in agreement, and if there is nothing, the address re-calculation section 4 will calculate the new address for registration. And registration is performed by the new address obtained by said count. This activity is done until the vacant entry is found (S5).

[0016] In addition, although it needs to be processed for a table 7 in fact in case there is no opening, in this example, since said processing part is easy, it is omitted, and is explained as what has an opening in a table 7. In said open address method, a collision may take place again at the calculated new address. In such a case, count of a new address will be continued until a vacant entry is found. Henceforth, in order to distinguish the collision which takes place by the first address calculation, and the collision which takes place by such re-calculation, the former will be called "primary impact" and the latter will be called "secondary impact."

[0017] **3: Processing explanation of chain method hash equipment Drawing 18 reference drawing 18 is the processing flow chart of conventional chain method hash equipment. Hereafter, processing of chain method hash equipment is explained based on drawing 18 . In addition, S11-S21 show each processing step.

[0018] Hereafter, since it is easy, the case of the tree structure of dichotomy is explained. In this processing, if Key k is given from the exterior (S11), the address generating section 3 will calculate

Address a by Hash Function h based on the given key k (S12). Next, {table(a) as for which the entry of Address a is vacant judges whether it is empty ?} (S13), and if vacant, a control section 5 will register the record corresponding to Key k and it into the entry of Address a (S16), and will end registration..

[0019] However, when the entry of Address a is not vacant, a control section 5 compares and judges {key(a) =k?} in accordance with the key k to which the key registered into Address a was given (S14). Consequently, if in agreement, since a key will be registered, processing is ended. However, if the record which already has other keys in the entry of the address a to register is registered when a key is not in agreement namely, a key to register will be compared with the already registered key {k:key (a)} using a certain comparator (S15).

[0020] Consequently, when the key k is larger, {k>key (a)} follows the link for a large key. Moreover, when the key k is smaller, {k<key (a)} follows the link for a small key. or [that the link tied to the larger one is vacant when following the link for a large key at this time] -- {-- or [that the link tied to the smaller one is vacant as for greater-link (a) when judging whether it is empty ?} no (S18) and following the link for a small key] -- {-- less-link (a) judges whether it is empty ?} no (S17).

[0021] consequently, when it judges that a link is vacant and comes out by said S17 and S18, since it means that registration by the key is not made, it registers the record of the key k which is going to carry out current registration. That is, a new entry is created and registered (S20).

[0022] Moreover, if the entry already registered into the link for [said] following exists (S17, S18), the link of the still smaller one or the link of the larger one will be followed (S19, S21). And the key and k are compared again. The record corresponding to Key k and it which should be registered by repeating this will be registered into the suitable location in the tree structure.

[0023] **4: Explanation of the example 1 of an experiment ... Drawing 19 reference drawing 19 is the explanatory view of the conventional example 1 of an experiment. The example 1 of an experiment is an example which experimented using the open address method hash equipment shown in drawing 16 . In this experiment, it experimented using the data of illustration and the experimental result was obtained.

[0024] as said experimental data -- Key k -- a, aa, ab, abc, b, bc, cd, cc, bb, d, e, and .... it is 19 keys (character string) which consist of bcb(s), and magnitude (address number) of a table was set to 25. In this case, the count of primary impact was 9 times. And the re-calculation of the address at the time of a collision added 1

(address a+1 of a dimension), and the new address was generated by taking the remainder broken by table size (magnitude of a table). Said experimental result was as illustration and secondary impact was 78 times.

[0025] **5: Explanation of the example 2 of an experiment ... Drawing 20 reference drawing 20 is the explanatory view of the conventional example 2 of an experiment. The example 2 of an experiment is an example which experimented using the chain method hash equipment shown in B Fig. of drawing 16 . In this experiment, the same data as said example 1 of an experiment were used, and the experimental result was obtained.

[0026] namely, -- as experimental data -- said example 1 of an experiment -- the same -- Key k -- a, aa, ab, abc, b, bc, cd, cc, bb, d, e, and .... it is 19 keys (character string) which consist of bcb(s), and magnitude (address number) of a table was set to 25. In this case, the count of primary impact was 9 times. The depth of a dichotomy link was deep like illustration (there are many counts of dichotomy).

[0027]

[Problem(s) to be Solved by the Invention] The following technical problems occurred in the above conventional things.

(1) : if a collision does not take place, the entry will be found in hash equipment in one count of a Hash Function to any keys, and it will be a very efficient approach. However, compared with the magnitude of the key set K, aggravation of the effectiveness according [the address set A] to a collision since it is very small is not avoided. Therefore, although it is important in the design of efficient hashing equipment how a collision (primary collision [secondary]) is prevented, it is difficult to reduce the count of a collision with said conventional equipment.

[0028] (2) : the number of entries of a table (the number of elements of the address set A) is m, when n entries are vacant, ideally, it is count of the first Hash Function in the probability of n/m, and a vacant entry will be found. However, usually it worsens rather than it because of the deviation of the data distribution which actually occurs. Moreover, in a hashing method, the secondary impact from which the count of a re-calculation of an address happens similarly for the deviation of data will be benefited large compared with ideal value.

[0029] This invention solves such a conventional technical problem, is using a virtual address effectively, lessens deviation of data, is decreasing a collision (especially secondary impact), and aims at realizing the high-speed hash equipment near the condition of an ideal. Moreover, when a key has a certain structure, this invention is using the

count equipment of the virtual key reflecting it, and aims at realizing not only improvement in the speed of access of associative retrieval equipment but flexibility of access.

[0030]

[Means for Solving the Problem] Drawing 1 is the principle explanatory view of this invention. This invention was constituted as follows in order to attain the aforementioned purpose.

: (1) The table 7 on which it had two or more entries which consist of a group of the record corresponding to one key and it in associative storage, and the address was shaken at each entry, The virtual key generating section 8 which generates a virtual key using a Hash Function based on the given key, Based on the virtual key generated in said virtual key generating section 8, it had the address generating section 3 which generates the address of said table, and access to said each entry was enabled with the address generated using the hashing method.

[0031] (2) : above (1) In associative storage, when an address is generated in said address generating section 3 and a collision takes place, it has the address re-calculation section 4 which re-calculates an address based on the virtual key generated in said virtual key generating section 8.

[0032] (3) : above (1) In associative storage, while establishing the link of two or more branching in each entry of said table 7, when an address is generated in said address generating section 3 and a collision takes place, it has the tree structure generation control means which generates the tree structure using said link by the size comparison of said virtual key.

[0033] : (4) The table on which it had two or more entries which consist of a group of one key, the standard key corresponding to it, and a record, and the address was shaken at each entry, The standard key generating section which generates a standard key using a Hash Function based on the given key, The virtual key generating section which generates a virtual key using a Hash Function based on the standard key generated in said standard key generating section, Based on the virtual key generated in said virtual key generating section, it had the address generating section which generates the address of said table, and access to said each entry was enabled with the address generated using the hashing method.

[0034] (Operation) The operation of this invention based on said configuration is explained based on drawing 1 .

(1) : above (1) As for the entry which consists of an operation key and a record, access is performed through two steps of actuation as follows. First, the virtual key generating section 8 will generate a virtual key by the Hash Function in the 1st step based on the key k, if Key k is

given. Next, in the 2nd step, the address generating section 3 generates the address in which the magnitude of a table more nearly actual than said obtained virtual key was made to reflect. Thus, if an address can be found, the address will perform access (registration/retrieval of data) to said entry.

[0035] Thus, the transformation method suitable for the purpose can be used for the virtual key generating section 8 and the address generating section 3 by dividing into two steps conventionally the address calculation processing which was being performed by once with one associative storage. It becomes possible to decrease a collision by generating few virtual keys of the bias to the key which the virtual key generating section 8 has a bias, and appears.

[0036] (2) : above (2) In the operation aforementioned associative storage, the address re-calculation section 4 re-calculates an address based on the virtual key generated in said virtual key generating section 8, when an address is generated in said address generating section 3 and a collision takes place.

[0037] In the re-calculation of said address, it is expectable that are using this, a new address with few biases will be easily calculated also in the case of a re-calculation, and its secondary impact decreases as a result since a virtual key appears uniformly.

[0038] (3) : above (3) In the operation aforementioned associative storage, a tree structure generation control means generates the tree structure by the size comparison of said virtual key using said link, when an address is generated in said address generating section 3 and a collision takes place.

[0039] In this case, the whole comparison is performed by the size comparison of a key only when a virtual key is in agreement. Since it is what abolished the bias of an appearance of a virtual key, if a link is a dichotomy tree, compared with the case where two keys are compared, about 1/of large and small incidence will be set to 2 in the comparison of a virtual key, for example.

[0040] Consequently, it is expected that the depth of the tree to near and the number of entries will become the minimum value to the tree which maintained the balance whose tree structure created is in an ideal condition mostly. Consequently, the count of retrieval of the tree structure for the registration to said entry and retrieval decreases, and it contributes to the improvement in effectiveness of processing.

[0041] (4) : above (4) The operation standard key generating section generates a standard key using a Hash Function based on the key given when the key was given, and the virtual key generating section generates a virtual key using a Hash Function based on the

standard key generated in said standard key generating section. Then, the address generating section generates the address of said table based on the virtual key generated in said virtual key generating section.

[0042] As mentioned above, a key is changed into standard form and it is used as a key of registration/retrieval without using the original key directly. If it does in this way, by use of a standard key, not only the retrieval by the original key but the retrieval by the standard key will become possible, and a kind of semantic retrieval will be attained. Moreover, a collision is reduced, and improvement in the speed of processing is enabled, and the flexibility of processing improves.

[0043] (5) : in addition to this, as mentioned above, by using a virtual key effectively, deviation of data can be lessened and high-speed associative storage (hash equipment) near the condition of an ideal can be realized by decreasing a collision (especially secondary impact). Moreover, when a key has a certain structure, not only improvement in the speed of access of associative storage but flexibility of access can be realized by using the virtual key generating section reflecting it.

[0044]

[Embodiment of the Invention] Hereafter, the gestalt of implementation of invention is explained to a detail based on a drawing. In addition, the following explanation describes the equipment using hashing technique "hash equipment." Moreover, although the control section for carrying out control based on hashing technique to said all examples of equipment is prepared, the illustration abbreviation has been carried out except for some examples of equipment. Furthermore, it is the virtual key k1 by Hash Function h from the given key k. The address to which the variable a for which it asked for and asked based on this virtual key k1 with a certain function (function Hash Function h and of the same kind) ad (k1) points is only described as "Address a."

[0045] **1: Explanation of the various examples of equipment ... For the explanatory view (A Fig. is the example 1 of equipment, and B Fig. is the example 2 of equipment) of the examples 1 and 2 of equipment, and drawing 3 the explanatory view of the example 3 of equipment and drawing 4 are [drawing 2 - drawing 6 reference drawing 2 / the explanatory view of the example 5 of equipment and drawing 6 of the explanatory view of the example 4 of equipment and drawing 5] the explanatory views of the example 6 of equipment. Hereafter, the various examples of equipment are explained based on drawing 2 - drawing 6 .

[0046] (1) : explanation of the example 1 of equipment ... The example 1 of

equipment shown in A Fig. of A Fig. reference drawing 2 of drawing 2 is an example of the hash equipment using a virtual key. This equipment is equipped with the virtual key generating section 8, the address generating section 3, the table 7, and the control section 5, and the entry which becomes every address (address) from the record corresponding to one key and it makes said table 7 have corresponded.

[0047] Although control of a control section 5 performs registration processing of the data (data of a key + record) to each entry of a table 7, and retrieval processing with this equipment, the virtual key generating section 8 generates a virtual key by the Hash Function from the given key in this case (generating), and the address generating section 3 generates an address (address) based on that virtual key. And each entry of a table 7 is accessed at said generated address, and registration/retrieval processing of data is performed.

[0048] (2) : explanation of the example 2 of equipment ... The example 2 of equipment shown in B Fig. of B Fig. reference drawing 2 of drawing 2 is an example of open address method hash equipment. This open address method hash equipment is equipped with the virtual key generating section 8, the address generating section 3, the address re-calculation section 4, the control section 5, and the table 7, and the entry which becomes every address (address) from the record corresponding to one key and it makes said table 7 have corresponded.

[0049] Although control of a control section 5 performs registration and retrieval processing of the data (data of a key + record) to each entry of a table 7 with this equipment, the virtual key generating section 8 generates a virtual key from the given key in this case (generating), and the address generating section 3 generates an address (address) based on that virtual key. And a table 7 is accessed with said generated address, and registration/retrieval processing of table data is performed.

[0050] In said processing, when the address generating section 3 calculates an address and primary impact happens, the address re-calculation section 4 calculates an address again by control of a control section 5 based on the virtual key generated in said virtual key generating section 8. And if said address re-calculation section 4 re-calculates an address and secondary impact happens again, the address re-calculation section 4 will do count of an address again further. Thus, a new address is calculated from a virtual key until a collision is lost, a table 7 is accessed with the new address, and registration/retrieval of data are performed.

[0051] (3) : explanation of the example 3 of equipment ... The example 3 of

equipment shown in drawing 3 reference drawing 3 is an example of virtual key hash equipment (this equipment is hereafter described as "chain method hash equipment") with the collision processor by the chain method. In this case, in order to simplify explanation, it is shown as an example of equipment in the case of a dichotomy tree.

[0052] This chain method hash equipment (in the case of a dichotomy tree) is equipped with the virtual key generating section 8, the address generating section 3, the control section (illustration abbreviation), and the table 7, and the entry which serves as one key, a virtual key corresponding to it, and a record from a dichotomy link makes said table 7 have corresponded to every address (address).

[0053] Although control of a control section performs registration and retrieval processing of the data (data of a key + virtual key + record + dichotomy link) to a table 7 with this equipment, a virtual key is generated in this case from the key to which the virtual key generating section 8 was given by the Hash Function (generating), and the address generating section 3 generates an address (address) based on that virtual key. And a table 7 is accessed at said generated address, and registration/retrieval processing of data is performed.

[0054] The tree structure is generated without calculating a new address, when primary impact happens in said processing. Therefore, the dichotomy link is established in each entry of a table 7. One is the link of the entry which registers a record with a bigger key than the key of the entry, and another side is a link to the entry of a small key. In the description of this equipment, the virtual key which was not based on the comparison of the original key currently performed conventionally as a comparison of a key, but was generated from the key will compare size.

[0055] (4) : explanation of the example 4 of equipment ... The example 4 of equipment shown in drawing 4 reference drawing 4 is an example of equipment which combined the example 2 (open address method hash equipment) of equipment, and the example 3 (chain method hash equipment) of equipment. This equipment is equipped with the virtual key generating section 8, the address generating section 3, the address re-calculation section 4, the control section (illustration abbreviation), and the table 7, and the entry which serves as one key, a virtual key corresponding to it, and a record from a dichotomy link makes said table 7 have corresponded to every address (address).

[0056] With this equipment, when primary impact happens, a vacant entry is looked for by calculating a new address using the re-calculation device (address

re-calculation section 4) in which a certain count is the description of said example 2 (open address method hash equipment) of equipment. When it is exceeded, an entry is secured using the link mechanism which is the description of the equipment of the example 3 (chain method hash equipment) of equipment, and it accesses there (registration/retrieval). When defining a certain fixed numbers beforehand and setting by count using a key, a virtual key, and an address, the case where it moves when specific conditions are satisfied, and various approaches are possible for the count for moving from a re-calculation to a link.

[0057] (5) : explanation of the example 5 of equipment ... The example 5 of drawing 5 reference equipment is associative storage (associative storage with standard key transformation) which changes the given key into standard form without using the original key directly, and uses it as a key of registration/retrieval. This equipment is equipped with the standard key generating section 12, the virtual key generating section 8, the address generating section 3, the control section (illustration abbreviation), and the table 7. And it enables it to have registered the key, the standard key, and the record into said table 7.

[0058] In the example 5 of equipment, the standard key generating section 12 generates a standard key from the given key, the virtual key generating section 8 generates a virtual key from said standard key, and the address generating section 3 generates an address from said virtual key. And a table is accessed using said address and registration/retrieval processing of data is performed.

[0059] (6) : explanation of the example 6 of equipment ... The example 6 of drawing 6 reference equipment is an example of equipment which combined the example 4 of equipment with the standard key generating section 12 of the example 5 of equipment. This equipment is equipped with the standard key generating section 12, the virtual key generating section 8, the address generating section 3, the control section (illustration abbreviation), and the table 7. And a key, the standard key corresponding to it, the virtual key, the record, and the dichotomy link are established in said table 7, and it enables it to have registered data into these.

[0060] In the example 6 of equipment, the standard key generating section 12 generates a standard key from the given key, the virtual key generating section 8 generates a virtual key from said standard key, and the address generating section 3 generates an address from said virtual key. And a table is accessed using said address and registration/retrieval processing of data is performed.

[0061] Moreover, when a collision occurs in said processing, it is constituted so

that the address re-calculation section 4 may re-calculate an address from said virtual key, may generate a new address and may perform access to a table 7.

[0062] **2: Processing explanation of the example 1 of equipment ... Drawing 7 reference drawing 7 is the processing flow chart of the example 1 of equipment. Hereafter, based on drawing 7 R> 7, the example of registration processing of the example 1 of equipment is explained. In addition, S51-S54 show each processing step. Moreover, the given key is set to k and k1 and an address are set to a for a virtual key.

[0063] In the example 1 of equipment, as for the entry which consists of one given key and record, access is performed through two steps of actuation as follows. First, when, as for the virtual key generating section 8, Key k is given in the 1st step (S51), it is the virtual key k1 by Hash-Function h (k) based on said given key k. It is made to generate (S52). Next, in the 2nd step, the address generating section 3 is said obtained virtual key k1. The address a in which the magnitude of an actual table was made to reflect is generated (S53). {a<-ad (k1)} In this case, said function ad (k1) is a function of the same class as said Hash Function h, and is the virtual key k1. The address (this is described as "Address a".) to which the variable a for which origin was asked with Function ad (k1) points is generated.

[0064] Thus, generating of Address a performs table processing (registration/retrieval processing of the data to the corresponding entry) with the address a (S54). Thus, the transformation method suitable for the purpose can be used for each equipment (the virtual key generating section 8, address generating section 3) by dividing into two steps conventionally the address calculation processing which was being performed by once with one hash equipment. The virtual key generating section 8 is the virtual key k1 which does not have the bias to the key k which has a bias and appears (few). The collision is decreased by generating. In addition, the magnitude of virtual key space is good to take to infinity or a sufficiently big finite number theoretically.

[0065] **3: Processing explanation of the example 2 of equipment ... Drawing 8 reference drawing 8 is the processing flow chart of the example 2 of equipment. Hereafter, based on drawing 8 R> 8, the example of registration processing of the example 2 of equipment is explained. In addition, S61-S67 show each processing step. Moreover, the given key is set to k and k1 and an address are set to a for a virtual key.

[0066] Also in the example 2 of equipment, access to the entry which consists of one given key and record is performed through two steps of actuation like said example 1 of equipment as follows. First, when, as for the virtual key

generating section 8, Key k is given in the 1st step (S61), it is the virtual key k1 by Hash Function h based on said given key k. It generates (S62). Next, in the 2nd step, the address generating section 3 is said obtained virtual key k1. The address a in which the magnitude of an actual table was made to reflect is generated (S63). {a<-ad (k1)}

[0067] Then, if a control section judges whether an opening is shown in the table of said address a (S64) and has an opening, it will register data (data of a key k+ record) into the entry (S66), and will end processing. However, by said processing of S64, when there is no opening in a table, {key(a) =k?} is judged [whether said given key k of a control section is the same as a registered key, and] (S65).

[0068] Consequently, although processing will be ended if the same, if not the same (primary impact occurs), it is said calculated virtual key k1 by the address re-calculation section 4. It is based and Address a is re-calculated (S67). {k1 <-next k1 and a<-ad (k1)} And it carries out repeatedly from processing of S64. In addition, it sets to said processing of S67, and is the virtual key k1. Key k to virtual key k1 again given when asking It calculates and is said {k1 <-next (k1, k)}, after that, and virtual key k1 which were calculated. It is also possible for it to be based and to re-calculate Address a.

[0069] In the example 2 of equipment, as explained above, when Address a is calculated by the aforementioned processing and primary impact happens there, the address re-calculation section 4 calculates a new address, and generates a new address. Also in a new address, when secondary impact happens, the address re-calculation section 4 generates a new address until a vacant entry is found.

[0070] It sets to the re-calculation (processing of S67) of an address, and the description of this equipment is the virtual key k1. It is in the point of using. By re-calculation processing of this address, it is the virtual key k1. It is also possible to use the calculated address a {k1 <-next (k1, k)}. Virtual key k1 Since it appears uniformly, it is using this, and a new address with few biases will be easily calculated also in the case of a re-calculation, and it is expected that secondary impact will decrease as a result.

[0071] In addition, in conventional hashing equipment, since the new address is calculated based on the address directly calculated from the key, even if it re-calculates, possibility that secondary impact will happen again is high. Even if it performs the re-calculation in consideration of the key and address of this invention similar to equipment, there is an increment in generating of the secondary impact by the appearance bias of a key. The equipment of this invention is using the virtual key

which appears uniformly as a substitute of a key, and decreases generating of secondary impact.

[0072] **4: Processing explanation of the example 3 of equipment ... Drawing 9 reference drawing 9 is the processing flow chart of the example 3 of equipment. Hereafter, processing of the example 3 of equipment is explained based on drawing 9 R> 9. In addition, S71-S82 show each processing step.

[0073] Various idea **** are comparatively easy for the chain method in the example 3 of equipment, and explain processing of equipment to it by the example of the chain by the efficient tree structure of dichotomy. In this equipment, count of an address is performed like processing of said example 1 of equipment. And when primary impact happens, unlike processing of the example 2 of equipment, the tree structure is generated instead of calculating a new address.

[0074] Therefore, the link of dichotomy is established in the entry of hashing equipment. One is the link of the entry which registers a record with a bigger key than the key of the entry, and another side is a link to the entry of a small key. In the description of this equipment, the virtual key which was not based on the comparison of the original key currently performed conventionally as a comparison of a key, but was generated from the key will compare size.

[0075] In this case, the whole comparison is performed by the size comparison of a key only when a virtual key is in agreement. Since it is what abolished the bias of an appearance of a virtual key, compared with the case where two keys are compared, about 1/of large and small incidence is set to 2 in the comparison of a virtual key. Consequently, it is expected that the depth of the tree to near and the number of entries will become the minimum value to the tree which maintained the balance whose tree structure created is in an ideal condition mostly. Consequently, also in a chain method, the count of retrieval of the tree structure for registration and retrieval decreases, and it contributes to the improvement in effectiveness of processing.

[0076] In processing of the example 3 of equipment, access to the entry which consists of one given key and record is performed through two steps of actuation as follows. First, when, as for the virtual key generating section 8, Key k is given in the 1st step (S71), it is the virtual key k1 by Hash Function h based on said given key k. It generates (S72). {k1 <-h(k)} Next, in the 2nd step, the address generating section 3 is said obtained virtual key k1. The address a in which the magnitude of an actual table was made to reflect is generated (S73). {a<-ad (k1)}

[0077] Next, {table(a) as for which the

entry of Address a is vacant judges whether it is empty ?} (S74), and if vacant, a control section 5 will register the record corresponding to Key k and it into the entry of Address a (S76), and will end registration. However, when the entry of Address a is not vacant, a control section 5 compares and judges {key(a) =k?} in accordance with the key k to which the key {key (a)} registered into Address a was given (S75).

[0078] Consequently, if in agreement, since a key is registered, processing will be ended. However, it is key {k1 by which the control section is already registered into the table 7 with said virtual key k1 if the record which already has other keys in the entry of the address a to register is registered when a key is not in agreement namely. : Compare key(a)} using a certain comparator (S77).

[0079] Consequently, virtual key k1 When the direction is large, {k1 ≥key(a)} follows the link for a large key. Moreover, virtual key k1 When the direction is small, {k1 <key(a)} follows the link for a small key.

[0080] or [that the link tied to the larger one of a table 7 is vacant when following the link for a large key at this time] -- {-- or [that the link tied to the smaller one is vacant as for greater-link (a) when judging whether it is empty ?} no (S81) and following the link for a small key] -- {-- less-link (a) judges whether it is empty ?} no (S78).

[0081] consequently, when it judges that a link is vacant and comes out by said processing of S81 and S78, since it means that registration by the key is not made, it registers the record of the key k which is going to carry out current registration. That is, a new entry is created and registered (S80).

[0082] Moreover, if the entry already registered into the link for [said] following exists (S81, S78), the link of the still smaller one or the link of the larger one will be followed (S82, S79). And the record corresponding to Key k and it which should be again registered by repeating from said processing of S75 will be registered into the suitable location in the tree structure.

[0083] **5: Processing explanation of the example 4 of equipment ... Drawing 10 reference drawing 10 is the processing flow chart of the example 4 of equipment. Hereafter, processing of the example 4 of equipment is explained based on drawing 10 . In addition, S91-S104 show each processing step. The example 4 of equipment is equipment which combined the description of the open address hash equipment shown in the example 2 of equipment, and the description of the chain method hash equipment shown in the example 3 of equipment.

[0084] By processing of this example 4 of equipment, when primary impact happens, a vacant entry is looked for by calculating a new address using the

re-calculation device (address re-calculation section 4) in which a certain count is the description of the example 2 of equipment. When it is exceeded, an entry is secured using the link mechanism which is the description of the example 3 of equipment, and it registers there.

[0085] in this case, the count for moving from a re-calculation to a link -- **: -- the case where a certain fixed numbers are defined beforehand -- **: -- when setting by count using a key, a virtual key, and an address, the case where it moves when the conditions of **:specification are satisfied, and various approaches are possible. Specifically, it is as follows.

[0086] Also in the example 4 of equipment, access to the entry which consists of one given key and record is performed through two steps of actuation as follows. First, when, as for the virtual key generating section 8, Key k is given in the 1st step (S91), it is the virtual key k1 by Hash Function h based on said given key k. It generates (S92). Next, in the 2nd step, the address generating section 3 is said obtained virtual key k1. The address a in which the magnitude of an actual table was made to reflect is generated (S93). {a<-ad (k1)}

[0087] Then, if a control section judges whether an opening is shown in the table of said address a (S94) and has an opening, it will register data (key k+ record) into the entry (S96), and will end processing. However, by said processing of S94, when there is no opening in a table, {key(a) =k?} is judged [whether said given key k of a control section is the same as a registered key, and] (S95).

[0088] Consequently, if the same, processing will be ended, but if not the same, it will judge whether a control section should link (S97). Consequently, when [which should be linked] it is judged that it does not come out, it is said calculated virtual key k1 by the address re-calculation section 4. It is based and Address a is re-calculated (S98). {k1 <-next k1 and a<-ad (k1)} And it carries out repeatedly from processing of S94.

[0089] In addition, it sets to processing of S97 and is the virtual key k1. Key k to virtual key k1 again given when asking It calculates and is said {k1 <-next (k1, k)}, after that, and virtual key k1 which were calculated. It is also possible for it to be based and to re-calculate Address a.

[0090] When it is judged in said processing of S97 on the other hand that it should link, a control section 5 is said virtual key k1. Key {k1 already registered into the table 7 : Compare key(a)} using a certain comparator (S99). Consequently, virtual key k1 When the direction is large, {k1 >key(a)} follows the link for a large key. Moreover, virtual key k1 When the direction is small, {k1 <key(a)} follows the link for a small key.

[0091] or [that the link tied to the larger one of a table 7 is vacant when following

the link for a large key at this time] -- {-- or [that the link tied to the smaller one is vacant as for greater-link (a) when judging whether it is empty ?} no (S103) and following the link for a small key] -- {-- less-link (a) judges whether it is empty ?} no (S100).

[0092] consequently, when it judges that a link is vacant and comes out by said processing of S103 and S100, since it means that registration by the key is not made, it registers the record of the key k which is going to carry out current registration. That is, a new entry is created and registered (S102).

[0093] Moreover, if the entry already registered into the link for [said] following exists (S103, S100), the link of the still smaller one or the link of the larger one will be followed (S101, S104). And the record corresponding to Key k and it which should be again registered by repeating said processing of S99 will be registered into the suitable location in the tree structure.

[0094] **6: Processing explanation of the example 5 of equipment ... Drawing 11 reference drawing 11 is the processing flow chart of the example 5 of equipment. Hereafter, processing of the example 5 of equipment is explained based on drawing 11 . In addition, S111-S115 show each processing step. Processing of this example 5 of equipment explains retrieval of PROLOG language (Program Logic language) and a theorem for the method of use of the associative storage using a key with a certain structure as an example. The basic structure of the data in PROLOG language is a term. Said term is the tree structure expression of 'f (a)', 'g (a, b)', 'h(a, g (b, a))', etc. A variable can be used into said expression.

[0095] For example, the capital letter name 'X' part in 'f (X)' and 'h (X, g (b, X))' is a variable. The term of arbitration can be substituted for a variable. For example, if 'g (a, Y)' is substituted for the variable 'X' of 'f (X)', the term 'f (g, (a, Y))' will be acquired. The same expression is substituted for the same variable when substitution is performed.

[0096] For example, if 'a' is substituted for the variable 'X' of 'h (X, g (b, X))', 'h (a, g (b, a))' will be obtained. Thus, the variable is serving to show the location for substitution and it is not important for it what kind of identifier is used as a variable name. For example, 'f (X)', 'f (Y)', and 'h (X, g (b, X))' and 'h (Z, g (b, Z))' are expressions with which it carries out the same work although the appearance as an expression differs. In addition, said small letter like a, b, and c is a constant, and X and a capital letter like Y are variables.

[0097] by the way, the right -- the logical expression with which things were proved is called a theorem. The once proved theorem is stored in a database and using for other theorem proving is expected. Although appearance differs,

when it is the same contents substantially like the case where it is related with use of the same variable as the example of said PROLOG in the case of retrieval of whether a certain logical expression with which it is expected that it is a theorem is stored in the theorem database, I want to treat as the same thing in such a situation, rather than to treat it as a different thing.

[0098] the expression with which appearance differs -- parenchyma -- there is the approach of changing into standard form as one means to judge the same thing, and comparing. for example, the thing for which '(a**b) **c' is changed with '(a**c)**(b**c)' -- ' ... ** ... ** ... the formula of the standard form is obtained (**:AND, **:OR).

[0099] This processing changes a key into standard form, and uses it as a key of registration/retrieval without using the original key directly. First, if Key k is given (S111), the standard key generating section 12 will generate the standard key s by Hash Function h based on the key k (S112).

[0100] Next, the virtual key generating section 8 is the virtual key k1 by Hash Function h based on said standard key s. It generates (S113). Then, the address generating section 3 generates Address a with Function ad based on said virtual key k1 (S114). And a control section processes a table 7 with said generated address a (S115). (registration/retrieval processing to an entry)

[0101] As mentioned above, a key is changed into standard form and it is used as a key of registration/retrieval without using the original key directly. If it does in this way, by use of a standard key, not only the retrieval by the original key but the retrieval by the standard key will become possible, and a kind of semantic retrieval will be attained. Moreover, a collision is reduced and improvement in the speed of processing is enabled. Furthermore, when a key has a certain structure, not only improvement in the speed of access of associative storage but the flexibility of access can be raised by using the virtual key reflecting it.

[0102] **7: Processing explanation of the example 6 of equipment ... Drawing 12 reference drawing 12 is the processing flow chart of the example 6 of equipment. Hereafter, processing of the example 6 of equipment is explained based on drawing 12 . In addition, S121-S135 show each processing step. In this processing, first, if Key k is given (S121), the standard key generating section 12 will generate the standard key s by Hash Function sk based on that key k (S122).

[0103] Next, the virtual key generating section 8 is the virtual key k1 by Hash Function h based on said standard key s. It generates (S123). Then, the address generating section 3 generates Address a by Hash Function ad based on said virtual key k1 (S124). Then, if a control

section judges whether an opening is shown in the table of said address a (S125) and has an opening, it will register data (data of a key k+ record) into the entry (S127), and will end processing. In addition, also when the standard key s is registered according to an application, it thinks.

[0104] However, by said processing of S125, when there is no opening in a table, {key(a) =k?} is judged [whether said given key k of a control section is the same as a registered key, and] (S126). Consequently, if the same, processing will be ended, but if not the same, it will judge whether a control section should link (S128).

[0105] Consequently, when [which should be linked] it is judged that it does not come out, it is said calculated virtual key k1 by the address re-calculation section 4. It is based and Address a is re-calculated (S129). {k1 <-next k1 and a<-ad (k1)} And it carries out repeatedly from processing of S125. In addition, it sets to said processing of S129, and is the virtual key k1. Key k to virtual key k1 again given when asking It calculates and is said {k1 <-next (k1, k)}, after that, and virtual key k1 which were calculated. It is also possible for it to be based and to re-calculate Address a.

[0106] When it is judged in said processing of S128 on the other hand that it should link, a control section 5 is said virtual key k1. Key {k1 already registered into the table 7 : Compare key(a)} using a certain comparator (S130). Consequently, virtual key k1 When the direction is large, {k1 >key(a)} follows the link for a large key. Moreover, virtual key k1 When the direction is small, {k1 <key(a)} follows the link for a small key.

[0107] or [that the link tied to the larger one of a table 7 is vacant when following the link for a large key at this time] -- {-- or [that the link tied to the smaller one is vacant as for greater-link (a) when judging whether it is empty ?} no (S134) and following the link for a small key] -- {-- less-link (a) judges whether it is empty ?} no (S131).

[0108] consequently, when it judges that a link is vacant and comes out by said processing of S131 and S134, since it means that registration by the key is not made, it registers the record of the key k which is going to carry out current registration. That is, a new entry is created and registered (S133).

[0109] Moreover, if the entry already registered into the link for [said] following exists (S131, S134), the link of the still smaller one or the link of the larger one will be followed (S132, S135). And the record corresponding to Key k and it which should be again registered by repeating said processing of S130 will be registered into the suitable location in the tree structure.

[0110] **8: Explanation of the example 1 of an experiment ... Drawing 13 reference

drawing 13 is the explanatory view of the example 1 of an experiment. The example 1 of an experiment is an example which experimented using the open address method hash equipment of the example 2 of equipment shown in drawing 2 . In this experiment, it experimented using the data of illustration and the experimental result was obtained.

[0111] as said experimental data -- Key k -- a, aa, ab, abc, b, bc, cd, cc, bb, d, e, and .... it is 19 keys (character string) which consist of bcb(s), and magnitude (address number) of a table was set to 25. In this case, the count of primary impact was 9 times. and -- however, the count of secondary impact became 18 times, and decreased sharply compared with the conventional example (the secondary impact of the conventional example was 78 times). The count of a collision decreased by this and it has been proved that improvement in the speed of processing was attained.

[0112] **5: Explanation of the example 2 of an experiment ... Drawing 14 reference drawing 14 is the explanatory view of the example 2 of an experiment. The example 2 of an experiment is an example which experimented using the chain method hash equipment of the example 3 of equipment shown in drawing 3 . In this experiment, the same data as said example 1 of an experiment were used, and the experimental result was obtained.

[0113] namely, -- as experimental data -- said example 1 of an experiment -- the same -- Key k -- a, aa, ab, abc, b, bc, cd, cc, bb, d, e, and .... it is 19 keys (character string) which consist of bcb(s), and magnitude (address number) of a table was set to 25. In this case, although the count of primary impact was 9 times, the depth of a dichotomy link was shallow like illustration compared with the conventional example (there are few counts of dichotomy). The count of a collision decreased by this and it has been proved that improvement in the speed of processing was attained.

[0114]

[Effect of the Invention] As explained above, according to this invention, there is the following effectiveness.

(1) : by using a virtual key effectively, deviation of data can be lessened and high-speed associative storage (hash equipment) near the condition of an ideal can be realized by decreasing a collision (especially secondary impact). Moreover, when a key has a certain structure, not only improvement in the speed of access of associative storage but the flexibility of access can be raised by using the virtual key reflecting it.

[0115] (2) : by using a virtual key, the count of a collision in the hash equipment (equipment which used hashing technique) which is a kind of associative storage can be decreased. Therefore, registration of the data to each entry and improvement in the speed of retrieval

processing can be attained.

[0116] (3) : by use of a standard key, not only the retrieval by the original key but the retrieval by the standard key is attained, and a kind of semantic retrieval is attained. Corresponding to each claim, there is the following effectiveness outside said effectiveness.

[0117] (4) : in claim 1, based on the key to which the virtual key generating section was given in the 1st step, generate a virtual key by the Hash Function and generate the address in which the address generating section made the magnitude of an actual table reflect from a virtual key in the 2nd step. Thus, the transformation method suitable for the purpose can be used for the virtual key generating section and the address generating section by dividing into two steps conventionally the address calculation processing which was being performed by once with one associative storage. It becomes possible to decrease a collision by generating few virtual keys of the bias to the key which the virtual key generating section has a bias and appears.

[0118] (5) : in claim 2, the address re-calculation section re-calculates an address based on a virtual key, when an address is generated in the address generating section and a collision takes place. In this case, since a virtual key appears uniformly in the re-calculation of an address, by using this, also in the case of a re-calculation, a new address with few biases will be calculated easily, and, as a result, can lessen secondary impact.

[0119] (6) : in claim 3, a tree structure generation control means generates the tree structure by the size comparison of a virtual key using a link, when an address is generated in the address generating section and a collision takes place. In this case, the whole comparison is performed by the size comparison of a key only when a virtual key is in agreement. Since it is what abolished the bias of an appearance of a virtual key, if a link is a dichotomy tree, compared with the case where two keys are compared, about 1/of large and small incidence will be set to 2 in the comparison of a virtual key, for example.

[0120] Consequently, it is expected that the depth of the tree to near and the number of entries will become the minimum value to the tree which maintained the balance whose tree structure created is in an ideal condition mostly. Consequently, the count of retrieval of the tree structure for the registration to said entry and retrieval decreases, and it contributes to the improvement in effectiveness of processing.

[0121] (7) : in claim 4, generate a standard key using a Hash Function based on the key to which the standard key generating section was given when the key was given, and the virtual key generating section generates a virtual key using a Hash Function based on a

standard key. Then, the address generating section generates an address based on a virtual key.

[0122] As mentioned above, a key is changed into standard form and it is used as a key of registration/retrieval without using the original key directly. If it does in this way, by use of a standard key, not only the retrieval by the original key but the retrieval by the standard key will become possible, and a kind of semantic retrieval will be attained. Moreover, a collision is reduced and improvement in the speed of processing is enabled. Furthermore, when a key has a certain structure, not only improvement in the speed of access of associative storage but the flexibility of access can be raised by using the virtual key reflecting it.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] It is the principle explanatory view of this invention.

[Drawing 2] It is the explanatory view of the examples 1 and 2 of equipment in the gestalt of operation.

[Drawing 3] It is the explanatory view of the example 3 of equipment in the gestalt of operation.

[Drawing 4] It is the explanatory view of the example 4 of equipment in the gestalt of operation.

[Drawing 5] It is the explanatory view of the example 5 of equipment in the gestalt of operation.

[Drawing 6] It is the explanatory view of the example 6 of equipment in the gestalt of operation.

[Drawing 7] It is the processing flow chart of the example 1 of equipment in the gestalt of operation.

[Drawing 8] It is the processing flow chart of the example 2 of equipment in the gestalt of operation.

[Drawing 9] It is the processing flow chart of the example 3 of equipment in the gestalt of operation.

[Drawing 10] It is the processing flow chart of the example 4 of equipment in the gestalt of operation.

[Drawing 11] It is the processing flow chart of the example 5 of equipment in the gestalt of operation.

[Drawing 12] It is the processing flow chart of the example 6 of equipment in the gestalt of operation.

[Drawing 13] It is the explanatory view of the example 1 of an experiment in the gestalt of operation.

[Drawing 14] It is the explanatory view of the example 2 of an experiment in the gestalt of operation.

[Drawing 15] It is the explanatory view 1 of the conventional example.

[Drawing 16] It is the explanatory view 2 of the conventional example.

[Drawing 17] It is the processing flow

chart of conventional open address method hash equipment.

[Drawing 18] It is the processing flow chart of conventional chain method hash equipment.

[Drawing 19] It is the explanatory view of the conventional example 1 of an experiment.

[Drawing 20] It is the explanatory view of the conventional example 2 of an experiment.

[Description of Notations]

3 Address Generating Section

4 Address Re-calculation Section

7 Table

8 Virtual Key Generating Section

12 Standard Key Generating Section

[Translation done.]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平10－49545

(43)公開日　平成10年(1998)２月20日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　17/30			Ｇ０６Ｆ　15/411	３１０	
Ｇ１１Ｃ　15/00			Ｇ１１Ｃ　15/00	Ｂ	
			Ｇ０６Ｆ　15/403	３５０Ｄ	

審査請求　未請求　請求項の数４　ＯＬ　（全 18 頁）

(21)出願番号　特願平8－206971

(22)出願日　平成８年(1996)８月６日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　南　俊朗
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

(74)代理人　弁理士　今村　辰夫　（外１名）

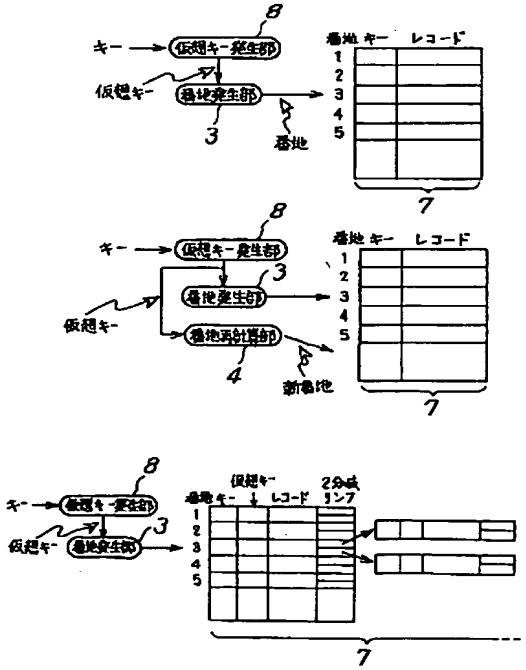
(54)【発明の名称】　連想記憶装置

(57)【要約】

【課題】本発明は連想記憶装置に関し、仮想キーを有効利用してデータの偏りを少なくし、衝突を減少させて理想の状態に近い高速な処理を可能とする。

【解決手段】１つのキーとレコードの組からなるエントリを複数持ち、各エントリに番地が振られたテーブル７と、キーを元にハッシュ関数を使用して仮想キーを発生させる仮想キー発生部８と、仮想キーを元にテーブルの番地を発生させる番地発生部３を備えた。また、番地を発生させて衝突が起こった場合、仮想キー発生部８で発生させた仮想キーを元に番地の再計算を行う番地再計算部４を備えた。更に、テーブル７の各エントリに複数分岐のリンクを設けると共に、番地を発生させて衝突が起こった場合、仮想キーの大小比較により、リンクを利用して木構造を生成する木構造生成制御手段を備えた。

本発明の原理説明図



1

【特許請求の範囲】

【請求項１】１つのキーとそれに対応したレコードの組からなるエントリを複数持ち、各エントリに番地が振られたテーブルと、与えられたキーを元にハッシュ関数を使用して仮想キーを発生させる仮想キー発生部と、前記仮想キー発生部で発生させた仮想キーを元に、前記テーブルの番地を発生させる番地発生部を備え、ハッシング法を用いて発生させた番地により前記各エントリへのアクセスを可能にしたことを特徴とする連想記憶装置。

【請求項２】前記番地発生部で番地を発生させて衝突が起こった場合、前記仮想キー発生部で発生させた仮想キーを元に、番地の再計算を行う番地再計算部を備えていることを特徴とした請求項１記載の連想記憶装置。

【請求項３】前記テーブルの各エントリに複数分岐のリンクを設けると共に、前記番地発生部で番地を発生させて衝突が起こった場合、前記仮想キーの大小比較により、前記リンクを利用して木構造を生成する木構造生成制御手段を備えていることを特徴とした請求項１記載の連想記憶装置。

【請求項４】１つのキーとそれに対応した標準キーとレコードの組からなるエントリを複数持ち、各エントリに番地が振られたテーブルと、与えられたキーを元にハッシュ関数を使用して標準キーを発生させる標準キー発生部と、前記標準キー発生部で発生させた標準キーを元にハッシュ関数を使用して仮想キーを発生させる仮想キー発生部と、前記仮想キー発生部で発生させた仮想キーを元に、前記テーブルの番地を発生させる番地発生部を備え、ハッシング法を用いて発生させた番地により前記各エントリへのアクセスを可能にしたことを特徴とする連想記憶装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、各種計算機に利用可能な連想記憶装置に関し、特にハッシング技法を利用した連想記憶装置に関する。連想記憶装置は、与えられたキーに対して、それに関連づけられたレコードを検索する装置であり、計算機言語処理系の記号テーブルの管理や、データベース等でのデータ連想検索が必要な装置一般に用いられている。

【０００２】

【従来の技術】以下、図面に基づき、従来例を説明する。

(1) ：連想記憶装置と、ハッシング装置の説明

図１５は従来例の説明図１であり、Ａ図は連想記憶装置、Ｂ図はハッシュ装置を示す。また、図１６は従来例の説明図２であり、Ａ図は開番地法ハッシュ装置、Ｂ図は連鎖法ハッシュ装置（２分岐木の場合）を示す。

【０００３】例えば、西原清一氏の解説「ハッシングの技法と応用」（情報処理学会誌、Ｖｏｌ，２１、Ｎｏ．９，Ｓｅｐ．１９８０）にも見られるように、計算機言

2

語の処理系で用いられているシンボルテーブルなどの連想記憶装置実現のための基本技術としてハッシング法を用いた装置（以下「ハッシング装置」、或いは「ハッシュ装置」と記す）は広く用いられている。

【０００４】連想記憶装置とは、図１５のＡ図に示したように、データに与えられた（通常は文字列である）キーｋを用いて、そのキーｋに関連づけられたデータ（以下「レコード」と記す）を検索する装置である。前記検索処理では、必要に応じて関連した情報が付加される。連想検索装置の概念は、キーｋとレコードを持ったテーブル（表）７として、表すことができる。

【０００５】例えば、計算機言語の処理系において、言語の解析を行うためには、プログラム中に現れた予約語、変数名などの識別子に対して、それが予約語であるか否かとか、変数名に対してその型は何かなどの様々な情報を格納し、利用する必要がある。このような状況において、識別子から、その関連情報を得るために連想記憶装置が用いられる。

【０００６】ところで、ハッシング装置は、連想検索装置の一種である。図１５のＢ図に示したように、ハッシング装置においては、連想記憶装置（テーブル７に対応）の各エントリ（１つのキーと、それに対応したレコードの組）に番地（アドレス）１、２、３、４、５・・・が振られている。例えば、ｍ個のエントリを持つテーブルの場合、０からｍ－１の番地（アドレス）が振られていると考えることができる。

【０００７】ハッシング装置における検索は、先ず、与えられたキーｋから、番地発生部３により、そのレコードを格納する番地が計算される。番地として得られる値は０以上ｍ未満でなければならない。キーｋから番地を計算する関数は、「ハッシュ関数」と呼ばれる。キーｋとしてとり得る値を「キー集合」と呼び、Ｋで表す。番地としてとり得る集合を「番地集合」と呼び、Ａで表す。従って、ハッシュ関数（番地発生部３の内部機能に対応する）ｈは、ｈ：Ｋ→Ａという関数であることになる。

【０００８】ハッシュ関数を用いて計算された番地のエントリが空いているならば、そこにキーｋと、それに関連づけられたレコードが格納される。検索の際も、与えられたキーｋより同様の方法で番地計算し、そのキーｋを持つエントリを探す。一般に、テーブルのエントリ数は、認識され得るキーの個数（集合Ｋの要素数で、理論的には無限の場合を含む）よりはるかに少ないため、異なったキーから同じ番地が計算される現象が起こる。これは、「衝突」と呼ばれる。

【０００９】前記衝突が起こった場合の対処として幾つかの方法が提案されている。その方法としては、開番地法、或いは連鎖法と呼ばれる方法などである。前記開番地法を利用した開番地法ハッシュ装置は、図１６のＡ図に示してあり、前記連鎖法を利用した連鎖法ハッシュ装

置は図１６のＢ図に示してある。

【００１０】前記開番地法ハッシュ装置では、番地発生部３が与えられたキーｋからテーブル７の番地を計算し、衝突が起こると、制御部５の指示により番地再計算部４が、前記計算された番地を元に、新たな番地を計算する方法である。この新たな番地の計算では、通常は、元の番地に何らかの方法で増分値を計算し、それを加えて新たな番地を計算する場合が多い。

【００１１】また、前記連鎖法ハッシュ装置では、テーブル７の各エントリにリンク情報を加え、テーブル７の内、若しくは外に設けたエントリに繋ぐものである。すなわち、番地発生部３が与えられたキーｋから番地を計算し、衝突が起こると更に、制御部５の指示により前記キーｋを用いて前記リンクを繋ぐ。また、従来、前記開番地法と連鎖法を組み合わせるやり方も提案されていた。

【００１２】§２：開番地法ハッシュ装置の処理説明・・・図１７参照

図１７は従来の開番地法ハッシュ装置の処理フローチャートである。以下、図１７に基づいて開番地法ハッシュ装置の処理を説明する。なお、Ｓ１～Ｓ６は各処理ステップを示す。

【００１３】この処理では、外部からキーｋが与えられると（Ｓ１）、番地発生部３は、与えられたキーｋを使用し、ハッシュ関数ｈ（ｋ）により変数ａを計算する。そして、番地発生部３は、前記変数ａが指し示す番地（以下、これを単に「番地ａ」と記す）を発生させる（Ｓ２）。

【００１４】次に制御部５は、番地ａのエントリが空いている｛ｔａｂｌｅ（ａ）は空？｝か否かを判断し（Ｓ３）、空いているならば、キーｋとそれに対応したレコードを番地ａのエントリに登録し（Ｓ６）、登録作業を終了する。しかし、番地ａのエントリが空いていない場合、制御部５は、番地ａに登録されたキーが与えられたキーｋと一致するか｛ｋｅｙ（ａ）＝ｋ？｝を比較して判断する（Ｓ４）。

【００１５】その結果、もし、一致するならば、キーは登録済みなので、処理は終了する。しかし、キーが一致しないならば、番地再計算部４は、登録のための新番地の計算を行う。そして、前記計算で得られた新番地によって、登録作業が行われる。この作業は、空いたエントリが見つかるまで行われる（Ｓ５）。

【００１６】なお、実際には、テーブル７に空きがない場合の処理が必要であるが、この例では、前記処理部分は簡単のため省略し、テーブル７に空きがあるものとして説明している。前記開番地法において、計算された新番地で再び衝突が起こる場合がある。そのような場合は、空きエントリが見つかるまで、新番地の計算を継続することになる。以降、最初の番地計算で起こる衝突と、このような再計算で起こる衝突を区別するために、

前者を「１次衝突」、後者を「２次衝突」と呼ぶことにする。

【００１７】§３：連鎖法ハッシュ装置の処理説明・・・図１８参照

図１８は従来の連鎖法ハッシュ装置の処理フローチャートである。以下、図１８に基づいて連鎖法ハッシュ装置の処理を説明する。なお、Ｓ１１～Ｓ２１は各処理ステップを示す。

【００１８】以下、簡単のために、２分岐の木構造の場合について説明する。この処理では、外部からキーｋが与えられると（Ｓ１１）、番地発生部３は、与えられたキーｋを元に、ハッシュ関数ｈにより番地ａを計算する（Ｓ１２）。次に制御部５は、番地ａのエントリが空いている｛ｔａｂｌｅ（ａ）は空？｝か否かを判断し（Ｓ１３）、空いているならば、キーｋとそれに対応したレコードを番地ａのエントリに登録し（Ｓ１６）、登録作業は終了する。

【００１９】しかし、番地ａのエントリが空いていない場合、制御部５は、番地ａに登録されたキーが与えられたキーｋと一致するか｛ｋｅｙ（ａ）＝ｋ？｝を比較して判断する（Ｓ１４）。その結果、もし、一致するならば、キーは登録済みなので、処理は終了する。しかし、キーが一致しない場合、すなわち、登録したい番地ａのエントリに、既に他のキーを持つレコードが登録されているならば、登録したいキーと、既に登録されているキー｛ｋ：ｋｅｙ（ａ）｝を何らかの比較機構を用いて比較する（Ｓ１５）。

【００２０】その結果、キーｋの方が大きい時｛ｋ＞ｋｅｙ（ａ）｝は、大きいキーのためのリンクを辿る。また、キーｋの方が小さい時｛ｋ＜ｋｅｙ（ａ）｝は、小さいキーのためのリンクを辿る。この時、大きいキーのためのリンクを辿る場合は、大きい方に繋ぐリンクが空いているか｛ｇｒｅａｔｅｒ－ｌｉｎｋ（ａ）は空？｝否かを判断（Ｓ１８）し、小さいキーのためのリンクを辿る場合は、小さい方に繋ぐリンクが空いているか｛ｌｅｓｓ－ｌｉｎｋ（ａ）は空？｝否かを判断（Ｓ１７）する。

【００２１】その結果、前記Ｓ１７、Ｓ１８でリンクが空きであると判断した時は、それは、そのキーでの登録がなされていないことを意味するので、現在登録しようとしているキーｋのレコードを登録する。すなわち、新エントリを作成し、登録する（Ｓ２０）。

【００２２】また、前記辿るためのリンクに既に登録されているエントリが存在するならば（Ｓ１７、Ｓ１８）、更に小さい方のリンク、或いは大きい方のリンクを辿る（Ｓ１９、Ｓ２１）。そして再び、そのキーとｋを比較する。これを繰り返すことで、登録されるべきキーｋと、それに対応したレコードは、木構造中の適切な場所に登録されることになる。

【００２３】§４：実験例１の説明・・・図１９参照

図１９は従来の実験例１の説明図である。実験例１は、図１６に示した開番地法ハッシュ装置を使用して実験を行った例である。この実験では、図示のデータを使用して実験を行い、実験結果を得た。

【００２４】前記実験データとしては、キーｋが、ａ，ａａ，ａｂ，ａｂｃ，ｂ，ｂｃ，ｃｄ，ｃｃ，ｂｂ，ｄ，ｅ，・・・・ｂｃｂからなる１９個のキー（文字列）であり、テーブルの大きさ（番地数）は２５とした。この場合、１次衝突の回数は９回であった。そして、衝突の時の番地の再計算は１を加え（元の番地ａ＋１）、テーブルサイズ（テーブルの大きさ）で割った余りをとることで新番地を発生させた。前記実験結果は図示の通りであり、２次衝突は７８回であった。

【００２５】§５：実験例２の説明・・・図２０参照

図２０は従来の実験例２の説明図である。実験例２は、図１６のＢ図に示した連鎖法ハッシュ装置を使用して実験を行った例である。この実験では、前記実験例１と同じデータを使用し、実験結果を得た。

【００２６】すなわち、実験データとしては、前記実験例１と同じように、キーｋが、ａ，ａａ，ａｂ，ａｂｃ，ｂ，ｂｃ，ｃｄ，ｃｃ，ｂｂ，ｄ，ｅ，・・・・ｂｃｂからなる１９個のキー（文字列）であり、テーブルの大きさ（番地数）は２５とした。この場合、１次衝突の回数は９回であった。２分岐リンクの深さは図示のように深かった（２分岐の回数が多い）。

【００２７】

【発明が解決しようとする課題】前記のような従来のものにおいては、次のような課題があった。

(1)：もし、衝突が起こらないならば、ハッシュ装置はどのようなキーに対しても、ハッシュ関数の１回の計算でそのエントリが見つかることになり、極めて効率の良い方法である。しかし、キー集合Ｋの大きさに比べて、番地集合Ａは極めて小さいため、衝突による効率の悪化は避けられない。従って、如何にして衝突（１次、及び２次等の衝突）を防ぐかが、効率の良いハッシング装置の設計に当たって重要であるが、前記従来の装置で衝突回数を減らすのは難しい。

【００２８】(2)：テーブルのエントリ数（番地集合Ａの要素数）がｍであり、ｎ個のエントリが空いているとき、理想的には、ｎ／ｍの確率で最初のハッシュ関数の計算で、空きエントリが見つかることになる。しかし、実際に生起するデータ分布の片寄りのために、それよりも悪くなるのが普通である。また、ハッシング法において、番地の再計算の回数も同様にデータの片寄りのために起こる２次衝突のために理想値と比べて大きくなることになる。

【００２９】本発明は、このような従来の課題を解決し、仮想番地を有効に用いることで、データの片寄りを少なくし、衝突（特に、２次衝突）を減少させることで、理想の状態に近い高速なハッシュ装置を実現させることを目的とする。また、本発明は、キーが或る構造を持つ時、それを反映した仮想キーの計算装置を用いることで、連想検索装置のアクセスの高速化のみならず、アクセスの柔軟性をも実現させることを目的とする。

【００３０】

【課題を解決するための手段】図１は本発明の原理説明図である。本発明は前記の目的を達成するため、次のように構成した。

(1)：連想記憶装置において、１つのキーとそれに対応したレコードの組からなるエントリを複数持ち、各エントリに番地が振られたテーブル７と、与えられたキーを元にハッシュ関数を使用して仮想キーを発生させる仮想キー発生部８と、前記仮想キー発生部８で発生させた仮想キーを元に、前記テーブルの番地を発生させる番地発生部３を備え、ハッシング法を用いて発生させた番地により前記各エントリへのアクセスを可能にした。

【００３１】(2)：前記(1)の連想記憶装置において、前記番地発生部３で番地を発生させて衝突が起こった場合、前記仮想キー発生部８で発生させた仮想キーを元に、番地の再計算を行う番地再計算部４を備えている。

【００３２】(3)：前記(1)の連想記憶装置において、前記テーブル７の各エントリに複数分岐のリンクを設けると共に、前記番地発生部３で番地を発生させて衝突が起こった場合、前記仮想キーの大小比較により、前記リンクを利用して木構造を生成する木構造生成制御手段を備えている。

【００３３】(4)：１つのキーとそれに対応した標準キーとレコードの組からなるエントリを複数持ち、各エントリに番地が振られたテーブルと、与えられたキーを元にハッシュ関数を使用して標準キーを発生させる標準キー発生部と、前記標準キー発生部で発生させた標準キーを元にハッシュ関数を使用して仮想キーを発生させる仮想キー発生部と、前記仮想キー発生部で発生させた仮想キーを元に、前記テーブルの番地を発生させる番地発生部を備え、ハッシング法を用いて発生させた番地により前記各エントリへのアクセスを可能にした。

【００３４】（作用）前記構成に基づく本発明の作用を、図１に基づいて説明する。

(1)：前記(1)の作用

キーとレコードからなるエントリは、次のように２段階の操作を経てアクセスが行われる。先ず、第１段階で、仮想キー発生部８は、キーｋが与えられると、そのキーｋを元にハッシュ関数により仮想キーを生成する。次に第２段階で、番地発生部３は、前記得られた仮想キーより、実際のテーブルの大きさを反映させた番地を発生する。このようにして番地が求まると、その番地により前記エントリへのアクセス（データの登録／検索）を行う。

【００３５】このように、従来、１つの連想記憶装置で、一度で行っていた番地計算処理を２段階に分けるこ

とで、仮想キー発生部８と、番地発生部３は、その目的に合った変換法を用いることができる。仮想キー発生部８は、偏りを持って出現するキーに対してその偏りの少ない仮想キーを生成することで衝突を減少させることが可能になる。

【００３６】(2) ：前記(2) の作用

前記連想記憶装置において、番地再計算部４は、前記番地発生部３で番地を発生させて衝突が起こった場合、前記仮想キー発生部８で発生させた仮想キーを元に、番地の再計算を行う。

【００３７】前記番地の再計算において、仮想キーは一様に出現するため、これを用いることで、再計算の際にも、偏りの少ない新番地が容易に計算されることになり、その結果２次衝突が少なくなることが期待できる。

【００３８】(3) ：前記(3) の作用

前記連想記憶装置において、木構造生成制御手段は、前記番地発生部３で番地を発生させて衝突が起こった場合、前記仮想キーの大小比較により、前記リンクを利用して木構造を生成する。

【００３９】この場合、仮想キーが一致した場合のみキーの大小比較によって全体の比較が行われる。仮想キーの出現の偏りを無くしたものなので、例えば、リンクが２分岐木であれば、２つのキーを比較する場合と比べ、仮想キーの比較においては、大小の出現率がほぼ１／２になる。

【００４０】その結果、作成される木構造は理想状態であるバランスのとれた木に近く、エントリ数に対する木の深さがほぼ最小値になることが期待される。その結果、前記エントリに対する登録、検索のための木構造の探索回数が減少し、処理の効率向上に貢献する。

【００４１】(4) ：前記(4) の作用

標準キー発生部は、キーが与えられると、与えられたキーを元にハッシュ関数を使用して標準キーを発生させ、仮想キー発生部は、前記標準キー発生部で発生させた標準キーを元にハッシュ関数を使用して仮想キーを発生させる。その後、番地発生部は、前記仮想キー発生部で発生させた仮想キーを元に、前記テーブルの番地を発生させる。

【００４２】前記のように、元のキーを直接使用しないで、キーを標準形に変換し、それを登録／検索のキーとして利用するものである。このようにすれば、標準キーの利用により、元のキーによる検索のみならず、標準キーによる検索も可能となり、一種の意味検索が可能になる。また、衝突を減らして処理の高速化を可能とし、かつ処理の柔軟性が向上する。

【００４３】(5) ：その他

以上のように、仮想キーを有効に用いることで、データの片寄りを少なくし、衝突（特に、２次衝突）を減少させることで、理想の状態に近い高速な連想記憶装置（ハッシュ装置）を実現させることができる。また、キーが或る構造を持つ時、それを反映した仮想キー発生部を用いることで、連想記憶装置のアクセスの高速化のみならず、アクセスの柔軟性をも実現させることができる。

【００４４】

【発明の実施の形態】以下、発明の実施の形態を図面に基づいて詳細に説明する。なお、以下の説明では、ハッシング技法を利用した装置を「ハッシュ装置」と記す。また、前記装置例の全てに、ハッシング技法に基づく制御を行うための制御部が設けてあるが、一部の装置例を除いて図示省略してある。更に、与えられたキーｋからハッシュ関数ｈにより仮想キーk_1を求め、この仮想キーk_1を元に或る関数（ハッシュ関数ｈと同種類の関数）ａｄ（k_1）により求めた変数ａが指し示す番地を、単に「番地ａ」と記す。

【００４５】§１：各種装置例の説明・・・図２～図６参照

図２は装置例１、２の説明図（Ａ図は装置例１、Ｂ図は装置例２）、図３は装置例３の説明図、図４は装置例４の説明図、図５は装置例５の説明図、図６は装置例６の説明図である。以下、図２～図６に基づき各種装置例について説明する。

【００４６】(1) ：装置例１の説明・・・図２のＡ図参照

図２のＡ図に示した装置例１は、仮想キーを利用したハッシュ装置の例である。この装置は、仮想キー発生部８と、番地発生部３と、テーブル７と、制御部５を備えており、前記テーブル７には、番地（アドレス）毎に、１つのキーとそれに対応したレコードからなるエントリが対応させてある。

【００４７】この装置では制御部５の制御によりテーブル７の各エントリへのデータ（キー＋レコードのデータ）の登録処理、及び検索処理を行うが、この場合、仮想キー発生部８が、与えられたキーからハッシュ関数により仮想キーを発生させ（生成し）、その仮想キーを元に、番地発生部３が番地（アドレス）を発生させる。そして、前記発生させた番地でテーブル７の各エントリをアクセスし、データの登録／検索処理を行う。

【００４８】(2) ：装置例２の説明・・・図２のＢ図参照

図２のＢ図に示した装置例２は開番地法ハッシュ装置の例である。この開番地法ハッシュ装置は、仮想キー発生部８と、番地発生部３と、番地再計算部４と、制御部５と、テーブル７とを備えており、前記テーブル７には、番地（アドレス）毎に、１つのキーとそれに対応したレコードからなるエントリが対応させてある。

【００４９】この装置では制御部５の制御によりテーブル７の各エントリへのデータ（キー＋レコードのデータ）の登録、及び検索処理を行うが、この場合、仮想キー発生部８が、与えられたキーから仮想キーを発生させ（生成し）、その仮想キーを元に、番地発生部３が番地

（アドレス）を発生させる。そして、前記発生させた番地によりテーブル７をアクセスし、テーブルデータの登録／検索処理を行う。

【００５０】前記処理において、番地発生部３が番地を計算し、１次衝突が起こった場合、制御部５の制御により、番地再計算部４は、前記仮想キー発生部８で発生させた仮想キーを元に、再度番地の計算を行う。そして、前記番地再計算部４が番地を再計算して再び２次衝突が起こったら、更に番地再計算部４が番地の計算をし直す。このようにして衝突がなくなるまで仮想キーから新番地の計算を行い、その新番地によりテーブル７をアクセスしてデータの登録／検索を行う。

【００５１】(3) ：装置例３の説明・・・図３参照

図３に示した装置例３は、連鎖法による衝突処理機構を持つ仮想キーハッシュ装置（以下、この装置を「連鎖法ハッシュ装置」と記す）の例である。この場合、説明を簡単にするため、２分岐木の場合の装置例として示してある。

【００５２】この連鎖法ハッシュ装置（２分岐木の場合）は、仮想キー発生部８と、番地発生部３と、制御部（図示省略）と、テーブル７とを備えており、前記テーブル７には、番地（アドレス）毎に、１つのキーと、それに対応した仮想キーと、レコードと、２分岐リンクからなるエントリが対応させてある。

【００５３】この装置では制御部の制御によりテーブル７へのデータ（キー＋仮想キー＋レコード＋２分岐リンクのデータ）の登録、及び検索処理を行うが、この場合、仮想キー発生部８がハッシュ関数により、与えられたキーから仮想キーを発生させ（生成し）、その仮想キーを基に、番地発生部３が番地（アドレス）を発生させる。そして、前記発生させた番地でテーブル７をアクセスし、データの登録／検索処理を行う。

【００５４】前記処理において１次衝突が起こった場合、新番地を計算しないで、木構造を生成する。そのためテーブル７の各エントリには２分岐リンクが設けてある。１つは、そのエントリのキーより大きなキーを持つレコードを登録するエントリのリンクであり、他方は、小さなキーのエントリへのリンクである。この装置の特徴は、キーの比較として、従来行われている元のキーの比較によるのではなく、キーより生成された仮想キーによって大小を比較しようというものである。

【００５５】(4) ：装置例４の説明・・・図４参照

図４に示した装置例４は、装置例２（開番地法ハッシュ装置）と装置例３（連鎖法ハッシュ装置）を組み合わせた装置例である。この装置は、仮想キー発生部８と、番地発生部３と、番地再計算部４と、制御部（図示省略）と、テーブル７とを備えており、前記テーブル７には、番地（アドレス）毎に、１つのキーと、それに対応した仮想キーと、レコードと、２分岐リンクからなるエントリが対応させてある。

【００５６】この装置では、１次衝突が起こった場合、或る回数までは前記装置例２（開番地法ハッシュ装置）の特徴である再計算機構（番地再計算部４）を用いて、新番地を計算することで、空きエントリを探す。それを越えた場合には、装置例３（連鎖法ハッシュ装置）の装置の特徴であるリンク機構を用いてエントリを確保し、そこにアクセス（登録／検索）するものである。再計算からリンクに移るための回数は、或る一定数を予め定めておく場合、キーや仮想キー、番地を用いた計算によって定める場合、特定の条件を満足した時に移る場合と様々な方法が可能である。

【００５７】(5) ：装置例５の説明・・・図５参照

装置例５は、元のキーを直接使用しないで、与えられたキーを標準形に変換し、それを登録／検索のキーとして利用する連想記憶装置（標準キー変換付き連想記憶装置）である。この装置は、標準キー発生部１２と、仮想キー発生部８と、番地発生部３と、制御部（図示省略）と、テーブル７を備えている。そして、前記テーブル７には、キーと、標準キーと、レコードを登録できるようにしてある。

【００５８】装置例５では、標準キー発生部１２が、与えられたキーから標準キーを発生させ、仮想キー発生部８が前記標準キーから仮想キーを発生させ、番地発生部３が前記仮想キーから番地を発生させる。そして、前記番地を使用してテーブルをアクセスし、データの登録／検索処理を行う。

【００５９】(6) ：装置例６の説明・・・図６参照

装置例６は、装置例５の標準キー発生部１２と、装置例４を組み合わせた装置例である。この装置は、標準キー発生部１２と、仮想キー発生部８と、番地発生部３と、制御部（図示省略）と、テーブル７を備えている。そして、前記テーブル７には、キーと、それに対応した標準キー、仮想キー、レコード、２分岐リンクが設けてあり、これらにデータを登録できるようにしてある。

【００６０】装置例６では、標準キー発生部１２が、与えられたキーから標準キーを発生させ、仮想キー発生部８が前記標準キーから仮想キーを発生させ、番地発生部３が前記仮想キーから番地を発生させる。そして、前記番地を使用してテーブルをアクセスし、データの登録／検索処理を行う。

【００６１】また、前記処理で衝突が発生した場合には、番地再計算部４が、前記仮想キーから番地の再計算を行い、新番地を生成してテーブル７へのアクセスを行うように構成されている。

【００６２】§２：装置例１の処理説明・・・図７参照

図７は装置例１の処理フローチャートである。以下、図７に基づいて装置例１の登録処理の例について説明する。なお、Ｓ５１～Ｓ５４は各処理ステップを示す。また、与えられたキーをｋ、仮想キーをk_1、番地をａとする。

【００６３】装置例１においては、与えられた１つのキーとレコードからなるエントリーは、次のように２段階の操作を経てアクセスが行われる。先ず、第１段階で、仮想キー発生部８は、キーｋが与えられると（Ｓ５１）、前記与えられたキーｋを元に、ハッシュ関数ｈ（ｋ）により仮想キーk_1を発生させる（Ｓ５２）。次に第２段階で、番地発生部３は、前記得られた仮想キーk_1より、実際のテーブルの大きさを反映させた番地ａを生成｛ａ←ａｄ（k_1）｝する（Ｓ５３）。この場合、前記関数ａｄ（k_1）は、前記ハッシュ関数ｈと同じ種類の関数であり、仮想キーk_1を元に関数ａｄ（ｋ１）により求めた変数ａが指し示す番地（これを「番地ａ」と記す。）を発生させる。

【００６４】このようにして番地ａが発生すると、その番地ａによりテーブル処理（該当するエントリへのデータの登録／検索処理）を行う（Ｓ５４）。このように、従来、１つのハッシュ装置で、一度で行っていた番地計算処理を２段階に分けることで、それぞれの装置（仮想キー発生部８、番地発生部３）は、その目的に合った変換法を用いることができる。仮想キー発生部８は、偏りを持って出現するキーｋに対してその偏りのない（少ない）仮想キーk_1を生成することで衝突を減少させている。なお、仮想キー空間の大きさは、原理的に無限、若しくは十分大きな有限数にとると良い。

【００６５】§３：装置例２の処理説明・・・図８参照
図８は装置例２の処理フローチャートである。以下、図８に基づいて装置例２の登録処理の例について説明する。なお、Ｓ６１～Ｓ６７は各処理ステップを示す。また、与えられたキーをｋ、仮想キーをk_1、番地をａとする。

【００６６】装置例２においても前記装置例１と同様に、与えられた１つのキーとレコードからなるエントリへのアクセスは、次のように２段階の操作を経て行われる。先ず、第１段階で、仮想キー発生部８は、キーｋが与えられると（Ｓ６１）、前記与えられたキーｋを元に、ハッシュ関数ｈにより仮想キーk_1を生成する（Ｓ６２）。次に第２段階で、番地発生部３は、前記得られた仮想キーk_1より、実際のテーブルの大きさを反映させた番地ａを生成｛ａ←ａｄ（k_1）｝する（Ｓ６３）。

【００６７】その後、制御部は、前記番地ａのテーブルに空きがあるか否かを判断し（Ｓ６４）、空きがあれば、そのエントリにデータ（キーｋ＋レコードのデータ）を登録して（Ｓ６６）、処理を終了する。しかし、前記Ｓ６４の処理でテーブルに空きがない場合、制御部は、前記与えられたキーｋが登録済みのキーと同じか否か｛ｋｅｙ（ａ）＝ｋ？｝を判断する（Ｓ６５）。

【００６８】その結果、同じであれば、処理を終了するが、同じでなければ（１次衝突が発生）、番地再計算部４により、前記計算した仮想キーk_1に基づいて番地ａの再計算｛k_1←ｎｅｘｔ　k_1，ａ←ａｄ（k_1）｝を行う（Ｓ６７）。そして、Ｓ６４の処理から繰り返して行う。なお、前記Ｓ６７の処理において、仮想キーｋ１を求める場合、再度与えられたキーｋから仮想キーｋ１を計算し｛k_1←ｎｅｘｔ（k_1，ｋ）｝、その後、前記計算した仮想キーk_1に基づいて番地ａの再計算を行うことも可能である。

【００６９】以上説明したように、装置例２では、前記の処理で番地ａを計算し、そこで１次衝突が起こった場合、番地再計算部４は新たな番地を計算し、新しい番地を発生させる。新番地においても、２次衝突が起こった場合、空きエントリが見つかるまで番地再計算部４は新しい番地を生成する。

【００７０】この装置の特徴は、番地の再計算（Ｓ６７の処理）において、仮想キーk_1を用いる点にある。この番地の再計算処理では、仮想キーk_1と共に、計算された番地ａを用いることも可能である｛k_1←ｎｅｘｔ（k_1，ｋ）｝。仮想キーk_1は一様に出現するため、これを用いることで、再計算の際にも、偏りの少ない新番地が容易に計算されることになり、その結果２次衝突が少なくなることが期待される。

【００７１】なお、従来のハッシング装置においては、キーから直接計算された番地を元に新番地を計算している為、再計算を行っても再び２次衝突が起こる可能性が高い。本発明の装置と類似のキーと番地を考慮した再計算を行ったとしても、キーの出現偏りによる２次衝突の発生の増加がある。本発明の装置は、一様に出現する仮想キーをキーの代わりとして用いることで、２次衝突の発生を減少させるものである。

【００７２】§４：装置例３の処理説明・・・図９参照
図９は装置例３の処理フローチャートである。以下、図９に基づいて装置例３の処理を説明する。なお、Ｓ７１～Ｓ８２は各処理ステップを示す。

【００７３】装置例３における連鎖法には様々考えられるが、比較的簡単で、効率の良い、２分岐の木構造による連鎖の例により、装置の処理を説明する。本装置において、番地の計算は、前記装置例１の処理と同様に行われる。そして、１次衝突が起こった場合、装置例２の処理と異なり、新番地を計算する代わりに木構造を生成する。

【００７４】そのため、ハッシング装置のエントリには２分岐のリンクが設けてある。１つは、そのエントリのキーより大きなキーを持つレコードを登録するエントリのリンクであり、他方は、小さなキーのエントリへのリンクである。この装置の特徴は、キーの比較として、従来行われている元のキーの比較によるのではなく、キーより生成された仮想キーによって大小を比較しようというものである。

【００７５】この場合、仮想キーが一致した場合のみキーの大小比較によって全体の比較が行われる。仮想キー

の出現の偏りを無くしたものなので、２つのキーを比較する場合と比べ、仮想キーの比較においては、大小の出現率がほぼ１／２になる。その結果、作成される木構造は理想状態であるバランスのとれた木に近く、エントリ数に対する木の深さがほぼ最小値になることが期待される。その結果、連鎖法においても、登録、検索のための木構造の探索回数が減少し、処理の効率向上に貢献する。

【００７６】装置例３の処理では、与えられた１つのキーとレコードからなるエントリへのアクセスは、次のように２段階の操作を経て行われる。先ず、第１段階で、仮想キー発生部８は、キーｋが与えられると（Ｓ７１）、前記与えられたキーｋを元に、ハッシュ関数ｈにより仮想キーk_1を生成｛$k_1 \leftarrow h(k)$｝する（Ｓ７２）。次に第２段階で、番地発生部３は、前記得られた仮想キーk_1より、実際のテーブルの大きさを反映させた番地ａを生成｛$a \leftarrow ad(k_1)$｝する（Ｓ７３）。

【００７７】次に制御部５は、番地ａのエントリが空いている｛ｔａｂｌｅ（ａ）は空？｝か否かを判断し（Ｓ７４）、空いているならば、キーｋとそれに対応したレコードを番地ａのエントリに登録し（Ｓ７６）、登録作業は終了する。しかし、番地ａのエントリが空いていない場合、制御部５は、番地ａに登録されたキー｛ｋｅｙ（ａ）｝が、与えられたキーｋと一致するか｛ｋｅｙ（ａ）＝ｋ？｝を比較して判断する（Ｓ７５）。

【００７８】その結果、もし、一致するならば、キーは登録済みなので処理は終了する。しかし、キーが一致しない場合、すなわち、登録したい番地ａのエントリに、既に他のキーを持つレコードが登録されているならば、制御部は、前記仮想キーk_1と、既にテーブル７に登録されているキー｛k_1：ｋｅｙ（ａ）｝を何らかの比較機構を用いて比較する（Ｓ７７）。

【００７９】その結果、仮想キーk_1の方が大きい時｛k_1＞ｋｅｙ（ａ）｝は、大きいキーのためのリンクを辿る。また、仮想キーk_1の方が小さい時｛k_1＜ｋｅｙ（ａ）｝は、小さいキーのためのリンクを辿る。

【００８０】この時、大きいキーのためのリンクを辿る場合は、テーブル７の大きい方に繋ぐリンクが空いているか｛ｇｒｅａｔｅｒ－ｌｉｎｋ（ａ）は空？｝否かを判断（Ｓ８１）し、小さいキーのためのリンクを辿る場合は、小さい方に繋ぐリンクが空いているか｛ｌｅｓｓ－ｌｉｎｋ（ａ）は空？｝否かを判断（Ｓ７８）する。

【００８１】その結果、前記Ｓ８１、Ｓ７８の処理でリンクが空きであると判断した時は、それは、そのキーでの登録がなされていないことを意味するので、現在登録しようとしているキーｋのレコードを登録する。すなわち、新エントリを作成し、登録する（Ｓ８０）。

【００８２】また、前記辿るためのリンクに既に登録されているエントリが存在するならば（Ｓ８１、Ｓ７８）、更に小さい方のリンク、或いは大きい方のリンク

を辿る（Ｓ８２、Ｓ７９）。そして再び、前記Ｓ７５の処理から繰り返すことで、登録されるべきキーｋと、それに対応したレコードは、木構造中の適切な場所に登録されることになる。

【００８３】§５：装置例４の処理説明・・・図１０参照

図１０は装置例４の処理フローチャートである。以下、図１０に基づいて装置例４の処理を説明する。なお、Ｓ９１～Ｓ１０４は各処理ステップを示す。装置例４は、装置例２に示した開番地ハッシュ装置の特徴と、装置例３に示した連鎖法ハッシュ装置の特徴とを組み合わせた装置である。

【００８４】この装置例４の処理では、１次衝突が起こった場合、或る回数までは装置例２の特徴である再計算機構（番地再計算部４）を用いて、新番地を計算することで、空きエントリを探す。それを越えた場合には、装置例３の特徴であるリンク機構を用いてエントリを確保し、そこに登録するものである。

【００８５】この場合、再計算からリンクに移るための回数は、①：或る一定数を予め定めておく場合、②：キーや仮想キー、番地を用いた計算によって定める場合、③：特定の条件を満足した時に移る場合と様々な方法が可能である。具体的には次の通りである。

【００８６】装置例４でも、与えられた１つのキーとレコードからなるエントリへのアクセスは、次のように２段階の操作を経て行われる。先ず、第１段階で、仮想キー発生部８は、キーｋが与えられると（Ｓ９１）、前記与えられたキーｋを元に、ハッシュ関数ｈにより仮想キーk_1を生成する（Ｓ９２）。次に第２段階で、番地発生部３は、前記得られた仮想キーk_1より、実際のテーブルの大きさを反映させた番地ａを生成｛$a \leftarrow ad(k_1)$｝する（Ｓ９３）。

【００８７】その後、制御部は前記番地ａのテーブルに空きがあるか否かを判断し（Ｓ９４）、空きがあれば、そのエントリにデータ（キーｋ＋レコード）を登録して（Ｓ９６）、処理を終了する。しかし、前記Ｓ９４の処理でテーブルに空きがない場合、制御部は、前記与えられたキーｋが登録済みのキーと同じか否か｛ｋｅｙ（ａ）＝ｋ？｝を判断する（Ｓ９５）。

【００８８】その結果、同じであれば、処理を終了するが、同じでなければ、制御部はリンクすべきか否かを判断する（Ｓ９７）。その結果、リンクすべきでないと判断した場合は、番地再計算部４により、前記計算した仮想キーk_1に基づいて番地ａの再計算｛$k_1 \leftarrow next\ k_1$，$a \leftarrow ad(k_1)$｝を行う（Ｓ９８）。そして、Ｓ９４の処理から繰り返して行う。

【００８９】なお、Ｓ９７の処理において、仮想キーk_1を求める場合、再度与えられたキーｋから仮想キーk_1を計算し｛$k_1 \leftarrow next(k_1, k)$｝、その後、前記計算した仮想キー$k_1$に基づいて番地ａの再計算を

行うことも可能である。

【００９０】一方、前記Ｓ９７の処理において、リンクすべきであると判断した場合、制御部５は、前記仮想キーk_1と、既にテーブル７に登録されているキー｛k_1：ｋｅｙ（ａ）｝を何らかの比較機構を用いて比較する（Ｓ９９）。その結果、仮想キーk_1の方が大きい時｛k_1＞ｋｅｙ（ａ）｝は、大きいキーのためのリンクを辿る。また、仮想キーk_1の方が小さい時｛k_1＜ｋｅｙ（ａ）｝は、小さいキーのためのリンクを辿る。

【００９１】この時、大きいキーのためのリンクを辿る場合は、テーブル７の大きい方に繋ぐリンクが空いているか｛ｇｒｅａｔｅｒ－ｌｉｎｋ（ａ）は空？｝否かを判断（Ｓ１０３）し、小さいキーのためのリンクを辿る場合は、小さい方に繋ぐリンクが空いているか｛ｌｅｓｓ－ｌｉｎｋ（ａ）は空？｝否かを判断（Ｓ１００）する。

【００９２】その結果、前記Ｓ１０３、Ｓ１００の処理でリンクが空きであると判断した時は、それは、そのキーでの登録がなされていないことを意味するので、現在登録しようとしているキーｋのレコードを登録する。すなわち、新エントリを作成し、登録する（Ｓ１０２）。

【００９３】また、前記辿るためのリンクに既に登録されているエントリが存在するならば（Ｓ１０３、Ｓ１００）、更に小さい方のリンク、或いは大きい方のリンクを辿る（Ｓ１０１、Ｓ１０４）。そして再び、前記Ｓ９９の処理を繰り返すことで、登録されるべきキーｋと、それに対応したレコードは、木構造中の適切な場所に登録されることになる。

【００９４】§６：装置例５の処理説明・・・図１１参照

図１１は装置例５の処理フローチャートである。以下、図１１に基づいて、装置例５の処理を説明する。なお、Ｓ１１１～Ｓ１１５は各処理ステップを示す。この装置例５の処理では、或る構造を持つキーを用いた連想記憶装置の利用の仕方をＰｒｏｌｏｇ言語（Program Logic言語）、定理の検索を例として説明する。Ｐｒｏｌｏｇ言語におけるデータの基本構造は項である。前記項は、例えば、‘ｆ（ａ）’，‘ｇ（ａ，ｂ）’，‘ｈ（ａ，ｇ（ｂ，ａ））’などの木構造表現である。前記表現内に変数を用いることができる。

【００９５】例えば、‘ｆ（Ｘ）’，‘ｈ（Ｘ，ｇ（ｂ，Ｘ））’の中の大文字名‘Ｘ’部分は変数である。変数には任意の項を代入できる。例えば、‘ｆ（Ｘ）’の変数‘Ｘ’に‘ｇ（ａ，Ｙ）’を代入すると、‘ｆ（ｇ，（ａ，Ｙ））’という項が得られる。代入が行われる時は、同じ変数には同じ表現が代入される。

【００９６】例えば、‘ｈ（Ｘ，ｇ（ｂ，Ｘ））’の変数‘Ｘ’に‘ａ’を代入すると、‘ｈ（ａ，ｇ（ｂ，ａ））’が得られる。このように、変数は、代入のための場所を示す働きをしており、変数名としてどのような名前を用いるかは重要ではない。例えば、‘ｆ（Ｘ）’と‘ｆ（Ｙ）’、そして、‘ｈ（Ｘ，ｇ（ｂ，Ｘ））’と‘ｈ（Ｚ，ｇ（ｂ，Ｚ））’は表現としての見かけは異なるが、同じ働きをする表現である。なお、前記ａ、ｂ、ｃのような小文字は定数であり、Ｘ、Ｙのような大文字は変数である。

【００９７】ところで、正しいことが、証明された論理式は定理と呼ばれる。一旦証明された定理は、データベースに格納し、他の定理の証明に用いることが期待されている。このような状況において、定理であることが予想される或る論理式が定理データベースに格納されているかの検索の際、前記Ｐｒｏｌｏｇの例と同様の変数の利用に関する場合と同様に見かけは異なるが、実質的には同じ内容である場合、それを異なるものとして扱うのではなく、同じものとして扱いたい。

【００９８】見かけの異なる表現が実質同じであることを判断する１つの手段として標準形へ変換して比べる方法がある。例えば、‘（ａ∧ｂ）∨ｃ’を、‘（ａ∨ｃ）∧（ｂ∨ｃ）’と変換することで、‘・・・∧・・・∧・・・’という標準形の式が得られる（∧：ＡＮＤ、∨：ＯＲ）。

【００９９】この処理は、元のキーを直接使用しないで、キーを標準形に変換し、それを登録／検索のキーとして利用するものである。先ず、キーｋが与えられると（Ｓ１１１）、標準キー発生部１２は、そのキーｋを元に、ハッシュ関数ｈにより標準キーｓを生成する（Ｓ１１２）。

【０１００】次に、仮想キー発生部８は前記標準キーｓを元に、ハッシュ関数ｈにより仮想キーk_1を生成する（Ｓ１１３）。その後、番地発生部３は、前記仮想キーk_1を元に、関数ａｄにより番地ａを生成する（Ｓ１１４）。そして、制御部は、前記生成した番地ａによりテーブル７の処理（エントリへの登録／検索処理）を行う（Ｓ１１５）。

【０１０１】前記のように、元のキーを直接使用しないで、キーを標準形に変換し、それを登録／検索のキーとして利用する。このようにすれば、標準キーの利用により、元のキーによる検索のみならず、標準キーによる検索も可能となり、一種の意味検索が可能になる。また、衝突を減らして処理の高速化を可能にできる。更に、キーが或る構造を持つ時、それを反映した仮想キーを用いることで、連想記憶装置のアクセスの高速化のみならず、アクセスの柔軟性をも向上させることができる。

【０１０２】§７：装置例６の処理説明・・・図１２参照

図１２は装置例６の処理フローチャートである。以下、図１２に基づいて、装置例６の処理を説明する。なお、Ｓ１２１～Ｓ１３５は各処理ステップを示す。この処理

では、先ず、キーｋが与えられると（Ｓ１２１）、標準キー発生部１２は、そのキーｋを元に、ハッシュ関数ｓｋにより標準キーｓを生成する（Ｓ１２２）。

【０１０３】次に、仮想キー発生部８は前記標準キーｓを元に、ハッシュ関数ｈにより仮想キーk_1を生成する（Ｓ１２３）。その後、番地発生部３は、前記仮想キーk_1を元に、ハッシュ関数ａｄにより番地ａを生成する（Ｓ１２４）。その後、制御部は前記番地ａのテーブルに空きがあるか否かを判断し（Ｓ１２５）、空きがあれば、そのエントリにデータ（キーｋ＋レコードのデータ）を登録して（Ｓ１２７）、処理を終了する。なお、用途に応じて標準キーｓも登録される場合も考えられる。

【０１０４】しかし、前記Ｓ１２５の処理でテーブルに空きがない場合、制御部は、前記与えられたキーｋが登録済みのキーと同じか否か｛ｋｅｙ（ａ）＝ｋ？｝を判断する（Ｓ１２６）。その結果、同じであれば、処理を終了するが、同じでなければ、制御部はリンクすべきか否かを判断する（Ｓ１２８）。

【０１０５】その結果、リンクすべきでないと判断した場合は、番地再計算部４により、前記計算した仮想キーk_1に基づいて番地ａの再計算｛k_1←ｎｅｘｔ（k_1，ａ←ａｄ（k_1））｝を行う（Ｓ１２９）。そして、Ｓ１２５の処理から繰り返して行う。なお、前記Ｓ１２９の処理において、仮想キーk_1を求める場合、再度与えられたキーｋから仮想キーk_1を計算し｛k_1←ｎｅｘｔ（k_1，ｋ）｝、その後、前記計算した仮想キーk_1に基づいて番地ａの再計算を行うことも可能である。

【０１０６】一方、前記Ｓ１２８の処理において、リンクすべきであると判断した場合、制御部５は、前記仮想キーk_1と、既にテーブル７に登録されているキー｛ｋ１：ｋｅｙ（ａ）｝を何らかの比較機構を用いて比較する（Ｓ１３０）。その結果、仮想キーk_1の方が大きい時｛k_1＞ｋｅｙ（ａ）｝は、大きいキーのためのリンクを辿る。また、仮想キーk_1の方が小さい時｛k_1＜ｋｅｙ（ａ）｝は、小さいキーのためのリンクを辿る。

【０１０７】この時、大きいキーのためのリンクを辿る場合は、テーブル７の大きい方に繋ぐリンクが空いているか｛ｇｒｅａｔｅｒ－ｌｉｎｋ（ａ）は空？｝否かを判断（Ｓ１３４）し、小さいキーのためのリンクを辿る場合は、小さい方に繋ぐリンクが空いているか｛ｌｅｓｓ－ｌｉｎｋ（ａ）は空？｝否かを判断（Ｓ１３１）する。

【０１０８】その結果、前記Ｓ１３１、Ｓ１３４の処理でリンクが空きであると判断した時は、それは、そのキーでの登録がなされていないことを意味するので、現在登録しようとしているキーｋのレコードを登録する。すなわち、新エントリを作成し、登録する（Ｓ１３３）。

【０１０９】また、前記辿るためのリンクに既に登録されているエントリが存在するならば（Ｓ１３１、Ｓ１３４）、更に小さい方のリンク、或いは大きい方のリンクを辿る（Ｓ１３２、Ｓ１３５）。そして再び、前記Ｓ１３０の処理を繰り返すことで、登録されるべきキーｋと、それに対応したレコードは、木構造中の適切な場所に登録されることになる。

【０１１０】§８：実験例１の説明・・・図１３参照
図１３は実験例１の説明図である。実験例１は、図２に示した装置例２の開番地法ハッシュ装置を使用して実験を行った例である。この実験では、図示のデータを使用して実験し、実験結果を得た。

【０１１１】前記実験データとしては、キーｋが、ａ，ａａ，ａｂ，ａｂｃ，ｂ，ｂｃ，ｃｄ，ｃｃ，ｂｂ，ｄ，ｅ，・・・・ｂｃｂからなる１９個のキー（文字列）であり、テーブルの大きさ（番地数）は２５とした。この場合、１次衝突の回数は９回であった。そして、しかし、２次衝突回数は１８回となり、従来例に比べて大幅に減少した（従来例の２次衝突は７８回であった）。これにより衝突回数が減少し、処理の高速化が可能になることが実証できた。

【０１１２】§５：実験例２の説明・・・図１４参照
図１４は実験例２の説明図である。実験例２は、図３に示した装置例３の連鎖法ハッシュ装置を使用して実験を行った例である。この実験では、前記実験例１と同じデータを使用し、実験結果を得た。

【０１１３】すなわち、実験データとしては、前記実験例１と同じように、キーｋが、ａ，ａａ，ａｂ，ａｂｃ，ｂ，ｂｃ，ｃｄ，ｃｃ，ｂｂ，ｄ，ｅ，・・・・ｂｃｂからなる１９個のキー（文字列）であり、テーブルの大きさ（番地数）は２５とした。この場合、１次衝突の回数は９回であったが、２分岐リンクの深さは、従来例に比べて図示のように浅かった（２分岐の回数が少ない）。これにより衝突回数が減少し、処理の高速化が可能になることが実証できた。

【０１１４】

【発明の効果】以上説明したように、本発明によれば次のような効果がある。

(1)：仮想キーを有効に用いることで、データの片寄りを少なくし、衝突（特に、２次衝突）を減少させることで、理想の状態に近い高速な連想記憶装置（ハッシュ装置）を実現させることができる。また、キーが或る構造を持つ時、それを反映した仮想キーを用いることで、連想記憶装置のアクセスの高速化のみならず、アクセスの柔軟性をも向上させることができる。

【０１１５】(2)：仮想キーを利用することで、連想記憶装置の一種であるハッシュ装置（ハッシング技法を使用した装置）における衝突回数を減少させることができる。そのため、各エントリに対するデータの登録、及び検索処理の高速化が達成可能である。

【０１１６】(3)：標準キーの利用により、元のキーに

よる検索のみならず、標準キーによる検索も可能になり、一種の意味検索が可能になる。前記効果の外、各請求項に対応して次のような効果がある。

【０１１７】(4)：請求項１では、第１段階で仮想キー発生部が与えられたキーを元に、ハッシュ関数により仮想キーを発生させ、第２段階で番地発生部が、仮想キーから、実際のテーブルの大きさを反映させた番地を発生させる。このように、従来、１つの連想記憶装置で、一度で行っていた番地計算処理を２段階に分けることで、仮想キー発生部と、番地発生部は、その目的に合った変換法を用いることができる。仮想キー発生部は、偏りを持って出現するキーに対してその偏りの少ない仮想キーを生成することで衝突を減少させることが可能になる。

【０１１８】(5)：請求項２では、番地再計算部は、番地発生部で番地を発生させて衝突が起こった場合、仮想キーを元に番地の再計算を行う。この場合、番地の再計算において仮想キーは一様に出現するため、これを用いることで、再計算の際にも、偏りの少ない新番地が容易に計算されることになり、その結果２次衝突を少なくすることができる。

【０１１９】(6)：請求項３では、木構造生成制御手段は、番地発生部で番地を発生させて衝突が起こった場合、仮想キーの大小比較により、リンクを利用して木構造を生成する。この場合、仮想キーが一致した場合のみキーの大小比較によって全体の比較が行われる。仮想キーの出現の偏りを無くしたものなので、例えば、リンクが２分岐木であれば、２つのキーを比較する場合と比べ、仮想キーの比較においては、大小の出現率がほぼ１／２になる。

【０１２０】その結果、作成される木構造は理想状態であるバランスのとれた木に近く、エントリ数に対する木の深さがほぼ最小値になることが期待される。その結果、前記エントリに対する登録、検索のための木構造の探索回数が減少し、処理の効率向上に貢献する。

【０１２１】(7)：請求項４では、標準キー発生部は、キーが与えられると、与えられたキーを元にハッシュ関数を使用して標準キーを発生させ、仮想キー発生部は、標準キーを元にハッシュ関数を使用して仮想キーを発生させる。その後、番地発生部は、仮想キーを元に番地を発生させる。

【０１２２】前記のように、元のキーを直接使用しないで、キーを標準形に変換し、それを登録／検索のキーとして利用する。このようにすれば、標準キーの利用により、元のキーによる検索のみならず、標準キーによる検索も可能となり、一種の意味検索が可能になる。また、衝突を減らして処理の高速化を可能にできる。更に、キーが或る構造を持つ時、それを反映した仮想キーを用いることで、連想記憶装置のアクセスの高速化のみならず、アクセスの柔軟性をも向上させることができる。

【図面の簡単な説明】

【図１】本発明の原理説明図である。

【図２】実施の形態における装置例１、２の説明図である。

【図３】実施の形態における装置例３の説明図である。

【図４】実施の形態における装置例４の説明図である。

【図５】実施の形態における装置例５の説明図である。

【図６】実施の形態における装置例６の説明図である。

【図７】実施の形態における装置例１の処理フローチャートである。

【図８】実施の形態における装置例２の処理フローチャートである。

【図９】実施の形態における装置例３の処理フローチャートである。

【図１０】実施の形態における装置例４の処理フローチャートである。

【図１１】実施の形態における装置例５の処理フローチャートである。

【図１２】実施の形態における装置例６の処理フローチャートである。

【図１３】実施の形態における実験例１の説明図である。

【図１４】実施の形態における実験例２の説明図である。

【図１５】従来例の説明図１である。

【図１６】従来例の説明図２である。

【図１７】従来の開番地法ハッシュ装置の処理フローチャートである。

【図１８】従来の連鎖法ハッシュ装置の処理フローチャートである。

【図１９】従来の実験例１の説明図である。

【図２０】従来の実験例２の説明図である。

【符号の説明】

３　番地発生部

４　番地再計算部

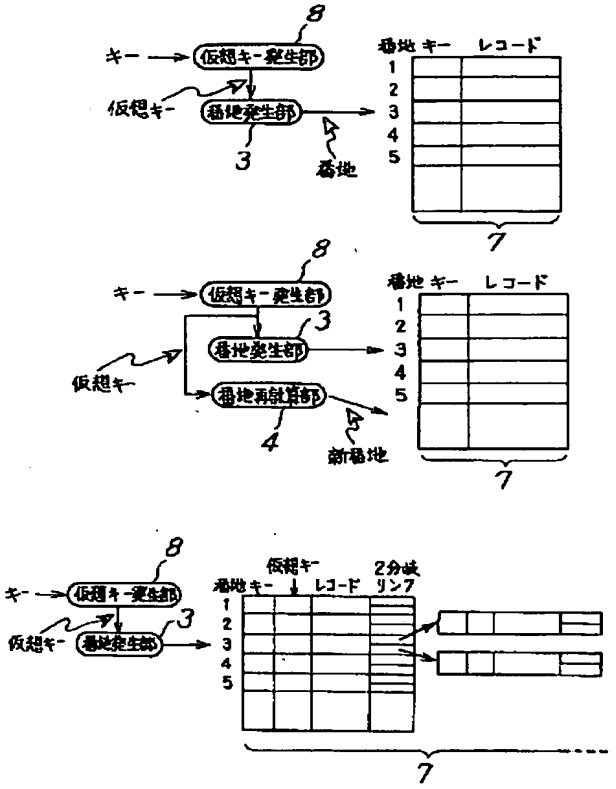
７　テーブル

８　仮想キー発生部

１２　標準キー発生部

【図1】

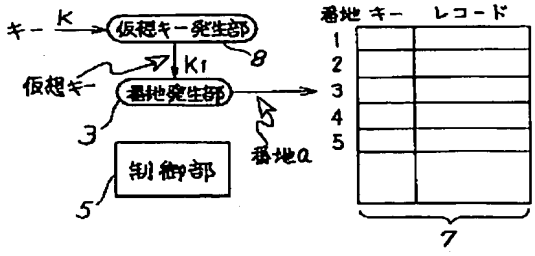
本発明の原理説明図



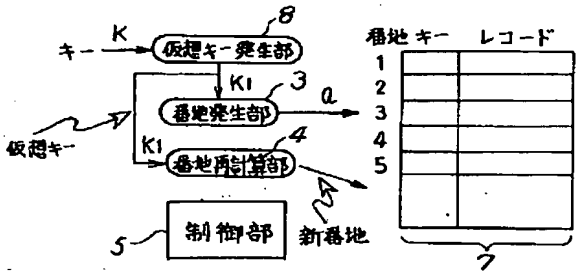
【図2】

装置例 1.2 の説明図

A：装置例 1（仮想キー利用ハッシュ装置）



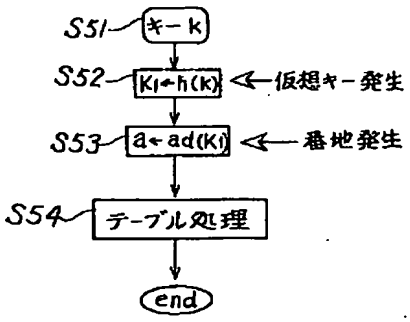
B：装置例 2（開番地法ハッシュ装置）



【図7】

装置例1の処理フローチャート

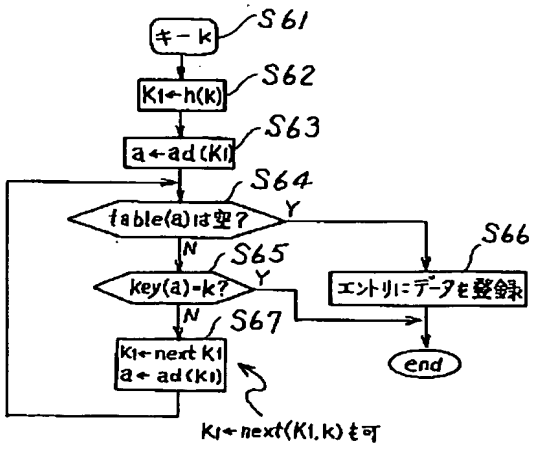
（仮想キー利用ハッシュ装置の処理）



【図8】

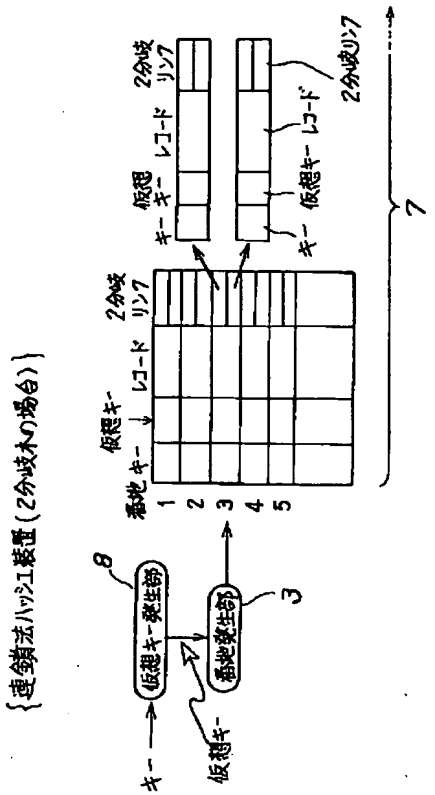
装置例2の処理フローチャート

（開番地法ハッシュ装置の処理）



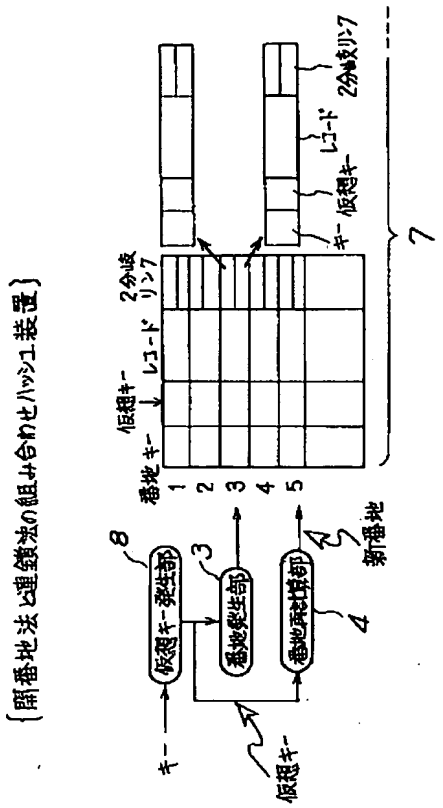
【図３】

装置例３の説明図



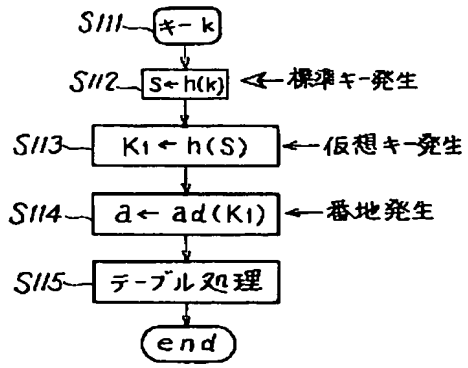
【図４】

装置例４の説明図



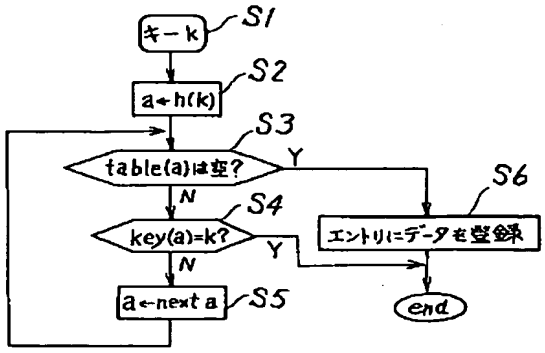
【図１１】

装置例５の処理フローチャート
（標準キー変換連想記憶装置の処理）



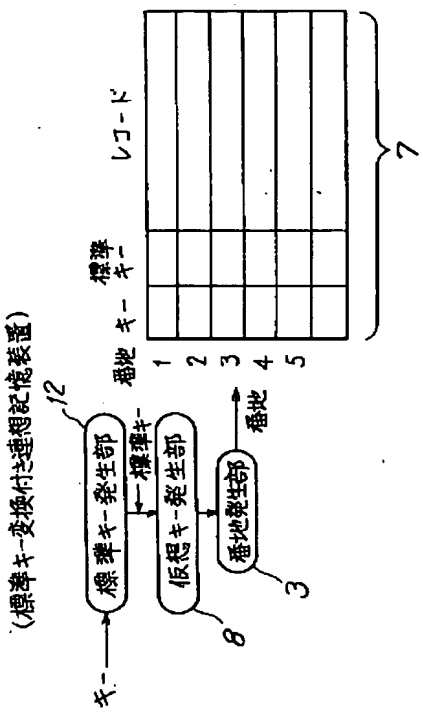
【図１７】

従来の開番地法ハッシュ装置の処理フローチャート



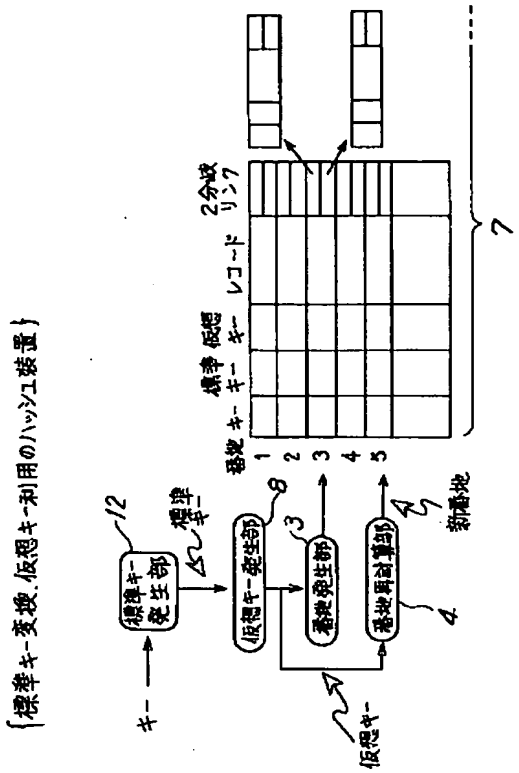
【図5】

装置例5の説明図



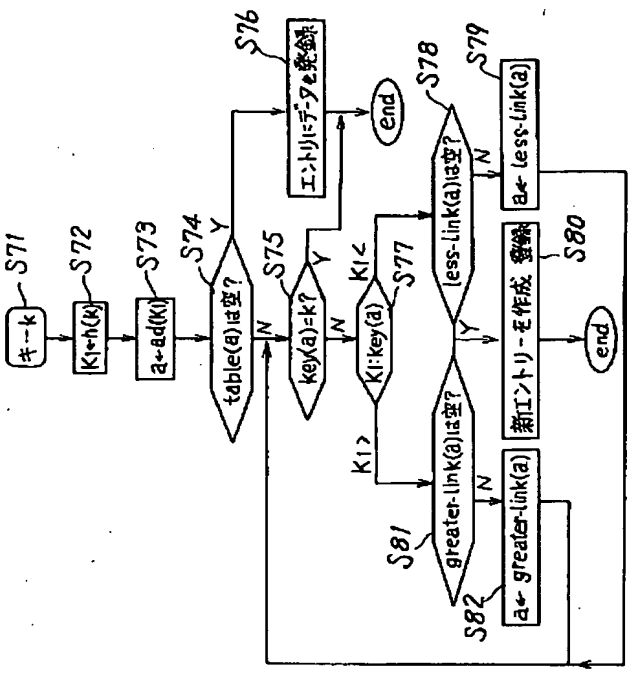
【図6】

装置例6の説明図



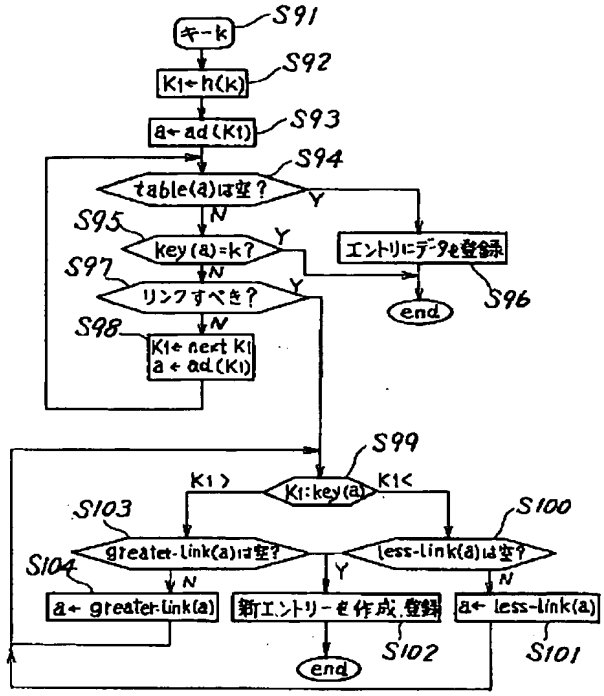
【図９】

装置例３の処理フローチャート
〔連鎖法ハッシュ装置の処理(2分岐木構造の場合)〕



【図１０】

装置例４の処理フローチャート
〔開番地法と連鎖法を組み合わせたハッシュ装置の処理
(２分岐木構造の場合)〕



【図１３】

実験例１の説明図

(仮想番地を用いた開番地法ハッシュ装置による結果)

キー: a, aa, ab, abc, b, bc, cd, cc, bb, d, e, ea, ba, bac, cba, abcdefg, bbb, ccc, bcb (19個)
テーブルの大きさ: 25
1次衝突の回数: 9

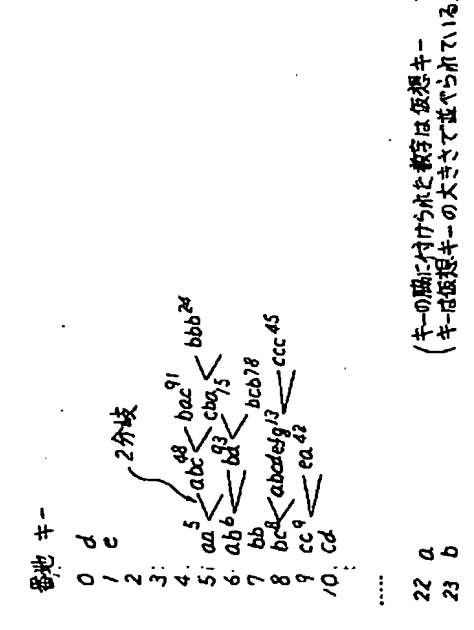
番地	キー
0	d
1	b
2	
3	bcb
4	cba
5	aa
6	ab
7	bb
8	bc
9	cc

番地	キー
10	cd
11	
12	
13	abcdefg
14	
15	e
16	bac
17	ea
18	ba
19	

番地	キー
20	ccc
21	
22	a
23	abc
24	bbb

2次衝突回数: 18

仮想番地を用いた連鎖法ハッシュ装置による結果

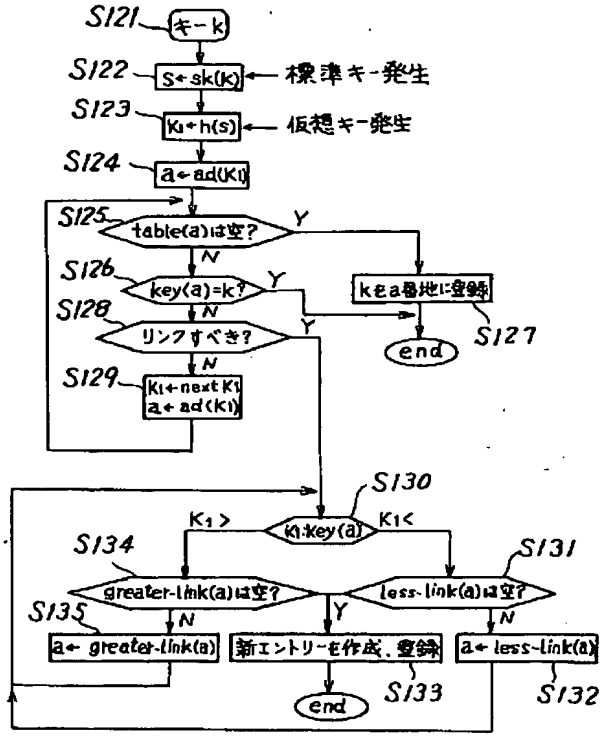


【図１４】

実験例２の説明図

【図１２】

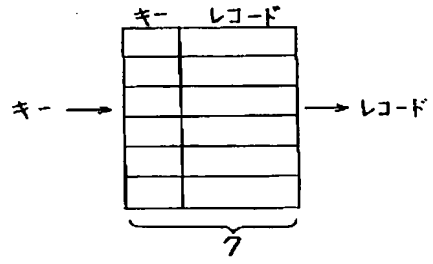
装置例６の処理フローチャート
（標準キー変換，仮想キー利用ハッシュ装置の処理）



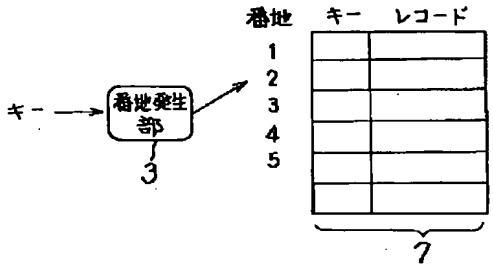
【図１５】

従来例の説明図１

A：連想記憶装置

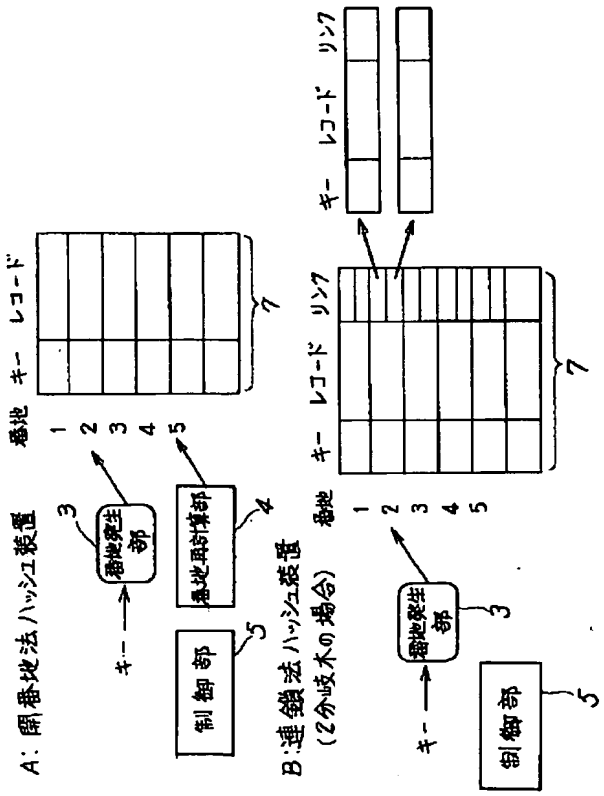


B：ハッシュ装置



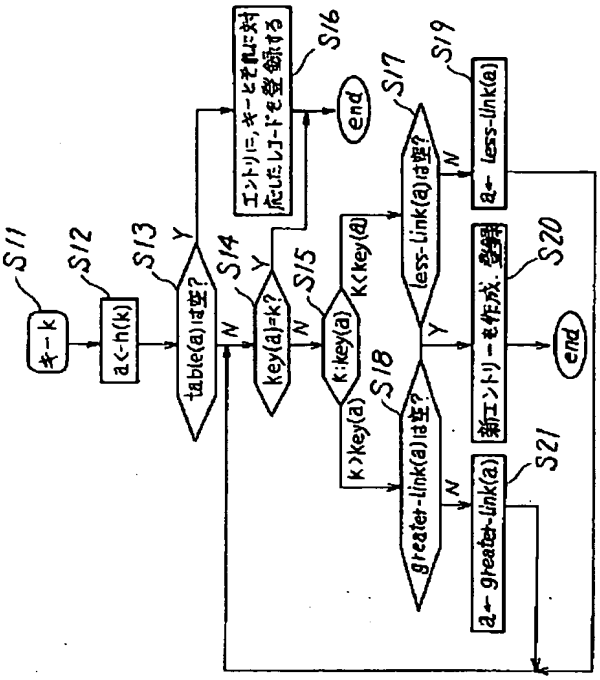
【図16】

従来例の説明図2



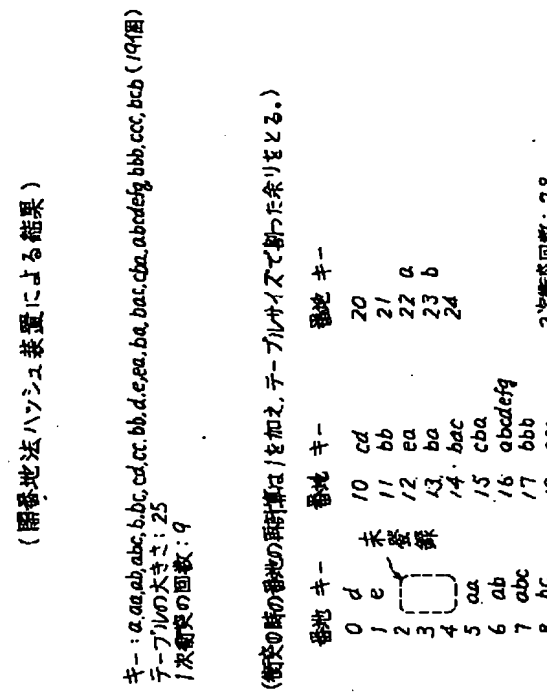
【図18】

従来の連鎖法ハッシュ装置の処理フローチャート



【図１９】

従来の実験例１の説明図



【図２０】

従来の実験例２の説明図